Revealing the impact of expertise on human planning with a two-player board game

Bas van Opheusden[1,2], Gianni Galbiati[1], Ionatan Kuperwajs[1], Zahy Bnaya[1], Yunqi Li[1], Wei Ji Ma[1]

[1]Center for Neural Science and Department of Psychology, New York University
[2]Department of Computer Science, Princeton University

Abstract

In recent years, artificial intelligence has made great progress in improving machine performance in tasks that require planning many steps ahead. By comparison, cognitive science has lagged behind in understanding human planning in complex tasks. One question of long-standing interest in this domain is whether skilled decision-makers plan further into the future than novices. Traditionally, the study of expertise in planning has focused on board games like chess, but the complexity of these games poses a barrier to detailed behavioral modeling. Conversely, common planning tasks in cognitive science are often lower-complexity and impose a ceiling for the depth to which any player can plan. Here, we investigate expertise in a complex board game that offers ample opportunity for skilled players to plan deeply. Despite this complexity, we show that human behavior can be captured using a computational cognitive model based on heuristic search. To validate this model, we predict human choices, response times, eye movements and perform a Turing test. Using the model, we find robust evidence for increased planning depth with expertise in both laboratory and large-scale mobile data. Our results highlight the promise of investigating human planning in complex tasks with precise behavioral modeling.

Introduction. Real-world decision-making often involves considering sequences of actions with multiple alternatives at each stage. Making such decisions requires people to mentally simulate the consequences of candidate actions multiple steps into the future using an internal model of the environment. This process is known as planning, and examples of ecologically relevant planning tasks are navigation, preparing a meal, career decisions or strategy games. Given the importance of planning to human behavior, a natural hypothesis is that skilled decision-makers are more successful because they plan further into the future.

Following seminal work by de Groot[1] and Simon & Chase[2], a growing body of literature has investigated the nature of expertise in planning by studying how expert chess players differ from less skilled counterparts[3, 4, 5]. This literature has explained the superior performance of experts as better pattern recognition[6, 7, 8] and/or deeper search[3, 9, 10, 11, 12]. However, developing computational cognitive models that accurately predict individual players' chess moves has proven difficult[13, 14], and instead studies tend to rely on clever experimental manipulations [15, 16] or verbal report analysis [17].

In contrast to the chess expertise literature, studies of planning in cognitive and neural science have often employed simpler tasks so that behavior and neural activity can be precisely modeled. These studies provide ample evidence that humans and animals engage in forward planning at decision time, and suggest candidates for the neural substrates supporting that behavior[18].

In humans, modeling people's choices in the classic two-step decision-making task[19] reveals a goal-directed planning component to their decision-making. In a more complex goal-directed decision-making task, Huys et al.[20, 21] found that people plan along multiple branches in a decision tree, but eliminate unpromising branches by pruning. Snider et al.[22] studied planning in a fast-paced, dynamic environment and found human behavior consistent with planning 3 to 4 steps into the future. Kolling et al.[23] demonstrated that people use prospective information to guide current choices, and located the representation of prospective information to cingulate and prefrontal cortices. Additionally, multi-step iterated reasoning has been studied in behavioral economics and game theory, resulting in concepts such as level-k reasoning[24] and the cognitive hierarchy[25].

In animals, signatures of prospective activity along possible trajectories an animal might take have been found in sequences of hippocampal place cell activity[26], particularly when an animal stops at a choice point[27]. Hippocampal neural activity has been associated with

both planning at decision time and planning in the background[28]. Additionally, evidence for planning in animals has been found in adaptations of the two-step task for rodents[29, 30, 31].

These human and animal studies rely on planning tasks of limited complexity, which impose a ceiling for the depth of planning and makes them less suitable to study the nature of expertise. The perfect task for studying expertise in planning needs to be complex enough that strong play requires thinking multiple moves ahead, but tractable for computational modeling. Additionally, to encourage learning, it should be novel, have simple rules, and be engaging. Here, we introduce a task that satisfies these competing desiderata, develop a computational cognitive model for human decision-making, validate it using choice, response time and eye movement data, and finally use the model to investigate the nature of expertise in planning.

Task. Our task is a generalization of tic-tac-toe, in which two players alternate placing tokens on a 4-by-9 board (Fig 1A); black moves first. The objective is to get four tokens in a row horizontally, vertically, or diagonally. The game, which we call 4-in-a-row, can be played online at `basvanopheusden.github.com`. With approximately $1.2 \cdot 10^{16}$ non-terminal states (section S2), this game has a state space complexity[32] that far exceeds tasks commonly used in cognitive science[33].

Model. We adapt our model of human planning from the artificial intelligence literature, in particular heuristic search[34, 35]. The core of a heuristic search algorithm is a *heuristic function*, which maps a given board state to a value estimate, often as a weighted linear combination of board features. For example, a common chess heuristic is to count pieces for both players, with different point values for different pieces (pawns, knights, rooks, etc). Similarly, our heuristic function counts how often particular features (Fig 1B) appear on the board. It weighs those counts by feature weights, resulting in a quick to compute but rough value estimate.

To refine the value estimate, the model explores a decision tree of possible continuations (Fig 1C,D). We base our model on best-first search[36]; this algorithm iteratively expands nodes

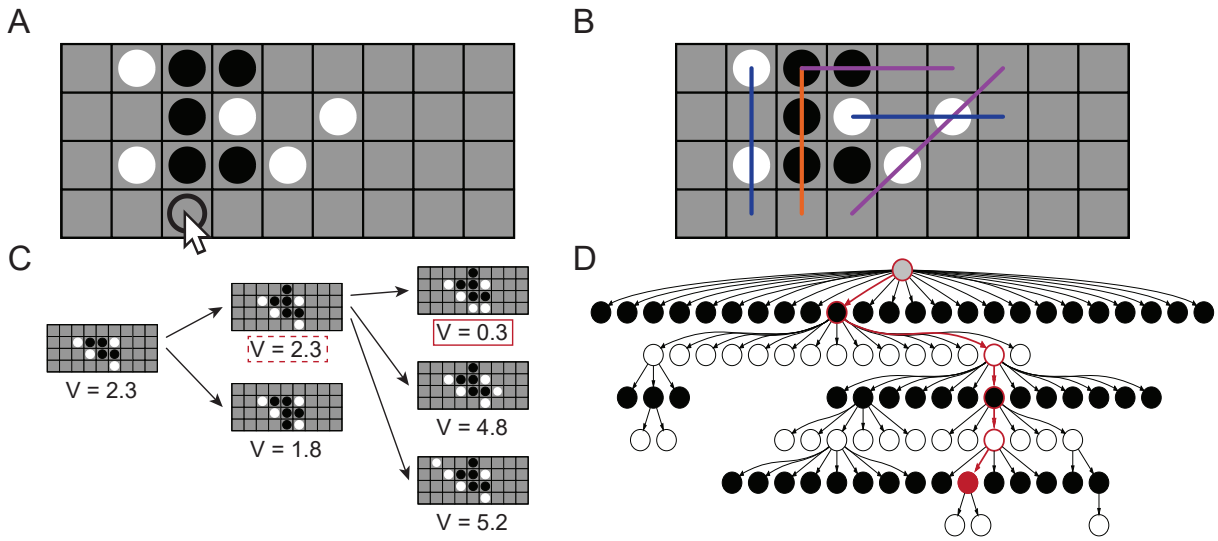

Figure 1: **Task and computational model. A.** Example board position in the 4-in-a-row game. Two players, black and white, alternate placing pieces on the board, and the first player to achieve 4-in-a-row wins the game. In this position, black is about to win by moving on the 3rd square in the bottom row (open circle, mouse cursor). **B.** Features used in the heuristic function. Features with identical colors are constrained to have identical weights. The model also includes a central tendency feature and a 4-in-a-row feature. **C.** Illustration of the heuristic search algorithm. In the root position (left), black is to move. After expanding the root node with two candidate moves for black and evaluating the resulting positions using $V(s)$, the algorithm selects the highest-value node ($V = 2.3$) on the second iteration and expands it with three candidate moves for white. The algorithm evaluates the resulting positions, and now backpropagates the lowest value ($V = 0.3$), since white is the opponent. That value will be compared against its alternatives in each intermediate node of the tree, to decide in which direction to expand the tree in the algorithm's next iteration. **D.** Decision tree built by the model with fitted parameters on an example board. The red nodes indicate the principal variation; the sequence of highest-value moves for both players. Note that different branches are evaluated to different depths.

on the principal variation, the sequence of actions that leads to the best outcome for both players given the current decision tree. Best-first search is particularly appealing as a human planning algorithm, since it effectively allocates computational resources to relevant branches of the decision tree, and such computational efficiency is a hallmark of human intelligence[37, 38, 39, 40].

Our other model choices are derived from cognitive science. Inspired by research from

Huys et al.[20, 21], the model prunes branches in the decision tree with low heuristic value. This improves the efficiency of search, but the model may fail to spot winning sequences if the starting move is low-value. Additionally, to allow the model to capture variability in human play and make human-like mistakes, we add Gaussian noise to the heuristic function and include feature dropout. For each move the model makes, it randomly omits some instances of features from the heuristic function before it performs search. We interpret these feature omissions cognitively as lapses of selective attention[41].

Model validation. We conducted several experiments and analyses to validate our computational model for human decision-making in 4-in-a-row. In our first experiment, 40 human participants play games against other human players, without any time pressure. For each participant, we estimate model parameters (feature weights, feature drop rate, decision tree size, pruning threshold and noise level) using 5-fold cross-validation. The model predicts out-of-sample choices with $40.8 \pm 1.4\%$ accuracy (mean and standard error across participants, two-sample T-test against chance: $t(39) = 26, p < 0.001$). Fig 2A shows an example model prediction, and Fig 2B the model accuracy for each participant. In sections S4-6, we show that the model's parameters can be reliably estimated with custom fitting methods[42, 43], and that it predicts multiple summary statistics. We validate our model specification by comparing against 22 alternative models, including ones that lesion model components. We find that a feature-based value function, tree search and a mechanism for attentional oversights are essential to predicting human choices (section S7). We select the main model as a parsimonious representative of that model class.

Next, we performed a "generalization" experiment in which 40 participants perform three tasks: playing against computer opponents, a two-alternative forced-choice (2AFC) between moves in a given position and a board evaluation task. We selected positions to make the decisions challenging, both to our participants and for the model to predict. For each player, we

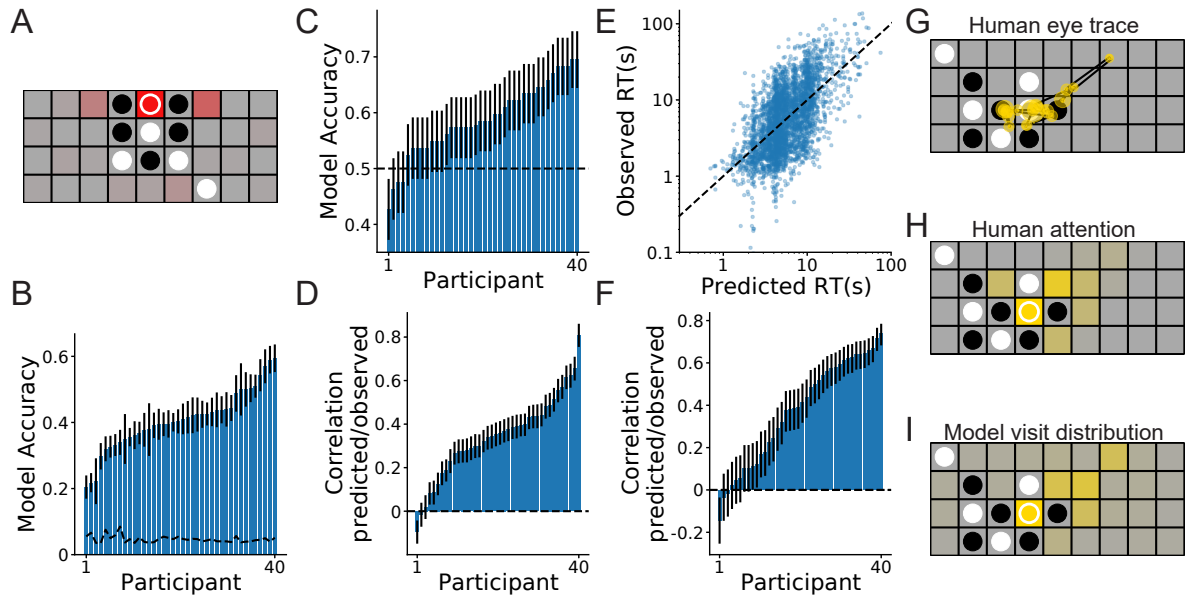

Figure 2: **The model accounts for multivariate data and generalizes to unseen data. A.** Example board position from a human-vs-human game. The white open circle indicates the move that the active player (white) chose. Red shading indicates the probability distribution of that participant's next move, as predicted by the model with parameters inferred for that participant using 5-fold cross-validation. **B.** Model accuracy (percent correctly predicted moves) for each participant in human-vs-human games, ranked from worst to best predicted. Error bars indicate standard error of the mean (s.e.m.) across all positions in that participant's games. The dashed line represents the accuracy of a "chance" model, which assumes that people move on a randomly selected unoccupied square. **C.** Model accuracy for 2AFC decisions in the generalization experiment. For each participant, we estimated model parameters from that participant's moves in games against computer opponents. **D.** Same as **C**, for the correlation between predicted and observed responses on the board evaluation task. **E.** Predicted and observed response times across all participants in the human-vs-human data. We exclude any positions with fewer than 6 or more than 30 pieces on the board. **F.** Correlation between predicted and observed response times for each participant, ranked from worst to best. **G.** Trajectory of eye movements on one example trial. The black lines represent saccades, the yellow circles fixations with duration proportional to the area of the circle. **H.** Estimated distribution of overt attention across unoccupied squares, obtained by convolving the eye trajectory with a Gaussian filter. **I.** Distribution of squares visited by the search algorithm, with model parameters estimated from the participant's choices.

estimate model parameters from their choices during human-vs-computer games, and predict their 2AFC and evaluation decisions. For both tasks, the model predicts people's choices above

chance (percent correct 2AFC: $p_{\text{correct}} = 58.6 \pm 1.0\%, t(39) = 8.3, p < 0.001$, Fig 2C, correlation predicted-observed evaluations: $\rho = 0.377 \pm 0.039, t(39) = 9.6, p < 0.001$, Fig 2D). These results suggest that the computational cognitive model can generalize between different choice tasks in the 4-in-a-row domain. In section S8, we show that the model's accuracy compares favorably to that of an oracle model (which makes objectively correct moves with random tie-breaking), suggesting that the model captures individual participants' subjective preferences.

Finally, we conducted a "Turing test" experiment[44], in which 30 observers, familiar with the game, decided whether sequences of moves, 9.38 on average, were generated by the model or by human players. Human observers were able to discriminate with only 55.4% accuracy, which suggests that the main model makes human-like decisions.

We tested the model's ability to predict process data by analyzing response times and eye movements. To predict a participant's response time on a single move, we amend the best-first search algorithm with an early-termination rule (section S10), which terminates search when the model's decision is unlikely to change with more iterations. With this modification, we can fit the model parameters on choice data, and predict response times as the mean size of the decision tree built by the model on each trial. Fig 2E shows the predicted and observed response times (in logarithmic space) across all participants in the human-vs-human experiment, Fig 2F the Pearson correlation for each participant ($\rho = 0.351 \pm 0.029, t(39) = 11.66, p < 0.001$).

To analyze eye movements, we conducted an experiment in which 10 participants played against computer opponents while we tracked their eye movements with an infra-red video-based eye tracker (section S11). Fig 2G shows one participant's fixation trajectory in an example board position. For each move made by each participant, we estimate the distribution of squares they overtly attend to by convolving their fixation trajectory with a Gaussian filter, truncating to unoccupied squares and averaging in time. We then show that this distribution is similar to the distribution of squares visited by the cognitive model during its search process (mean correlation

across participants: $\rho = 0.535 \pm 0.024, t(9) = 21, p < 0.001$, see Fig 2H,I). In section S11, we show that this correlation is driven by branches in the decision tree occasionally reaching up to 7 moves deep.

The ability of the model to predict both response times and eye movements on individual trials suggests that people plan their moves by building decision trees, using an algorithm similar to that of our cognitive model.

Expertise in planning. The cognitive model allows us to investigate how expert players differ from novices. To do so, we performed a learning experiment in which 30 participants play against computer opponents for 5 sessions, each spaced no more than 2 days apart. We measure participants' task performance using Elo ratings[45], with a common baseline across all experimental data (section S2.1).

In Fig 3A, we show that participants' playing strength increases during the 5 sessions of the learning experiment (linear regression: $\beta = 21 \pm 4.6, p < 0.001$). To investigate which aspects of people's decision-making process underlie this performance increase, we convert the set of parameters inferred for each participant in each session to 3 metrics: planning depth, feature drop rate and heuristic quality (section S2.7). The planning depth is defined as the length of the principal variation in the tree, and roughly corresponds to the number of steps one thinks ahead. The feature drop rate is defined as the attentional lapse probability, a model parameter. Finally, the heuristic quality is defined as the correlation between heuristic and objective value, and therefore measures the "correctness" of the feature weights. Thus, these metrics map to different hypotheses on the nature of expertise (see discussion). In section S4, we show that these metrics, and planning depth in particular, can be reliably inferred from choice data, and together they explain 56.7% of the variance in playing strength (section S12).

Fig 3B-D shows that planning depth increases across sessions ($\beta = 0.255 \pm 0.061, p < 0.001$) while feature drop rate decreases ($\beta = -0.0119 \pm 0.0028, p < 0.001$). Heuristic quality

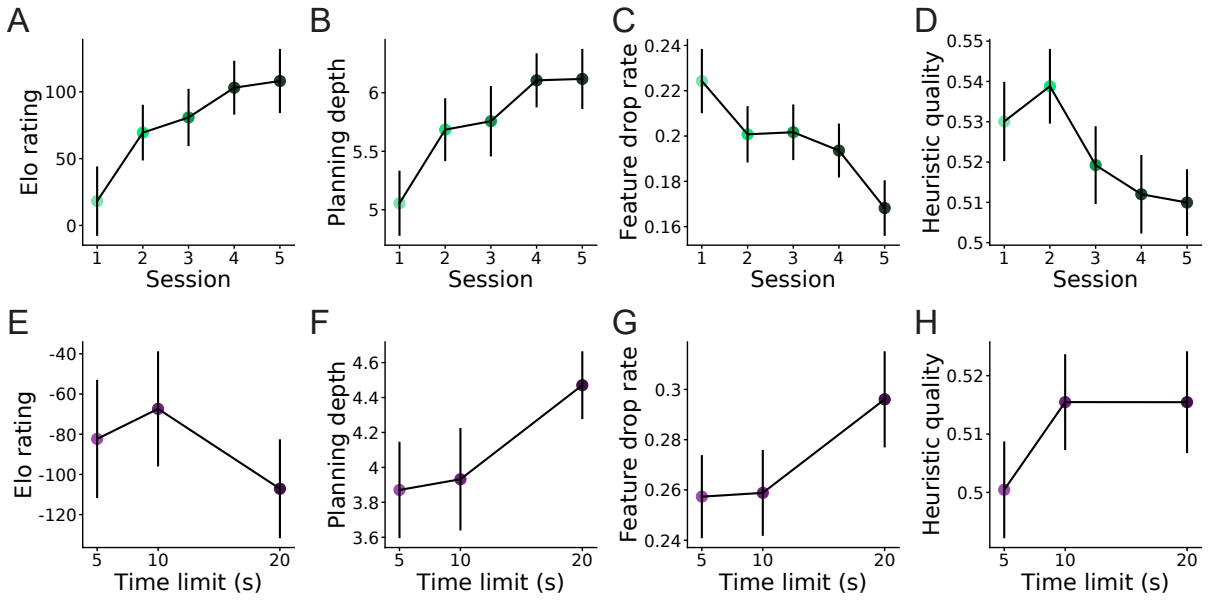

Figure 3: **The effects of expertise and time pressure on planning. A.** Average Elo rating of participants in the learning experiment, as a function of session number. In this panel and all others, error bars denote mean and standard error across participants. **B.** Average depth to which participants plan, as estimated by the behavioral model. **C.** Same as **B**, for feature drop rate. **D.** Same as **B**, for heuristic quality. **E.** Average Elo rating of participants in the time pressure experiment, as a function of the time limit. **F-H.** Same as **B-D**, for the time pressure experiment.

does not increase, and even decreases slightly ($\beta = -0.0067 \pm 0.0020, p = 0.0012$). In section S13, we show that individual differences in playing strength are also correlated with planning depth and feature drop rate, but not heuristic quality. Finally, in section S12, we break down the overall gain of 90 ± 26 Elo points between sessions 1 and 5 as a gain of 36 ± 11 points due to increased planning, a gain of 46 ± 12 due to attention, and a loss of 6.6 ± 3.5 points due to heuristic quality. These results suggest that stronger players plan deeper and have fewer lapses of attention. We find no evidence for improvements in feature weights.

In section S14, we show that participants' response time decreases across sessions, verifying that the planning depth increase is not a result of slower play. Another potential concern is that the estimates of planning depth, feature drop rate and heuristic quality are dependent on

our model specification, and human planning strategies may deviate from the model. Specifically, one may worry that people use features not present in our heuristic function, and that the model confuses increases in the weights of those feature across sessions with increased planning. In section S17, we investigate this possibility. First, we note that planning depth and feature weights are not confusable, at least for features in our model (section S4). Additionally, we show that the main result in Fig 3 is robust across all alternative model specifications. Although the existence of additional features in people's heuristic functions is impossible to rule out completely, we have no evidence suggesting that adding features to the model will change the main result of increased planning and improved attention with expertise.

Time pressure. To experimentally validate the planning depth metric, we conducted a time pressure experiment in which 30 participants played against computer opponents, with a time limit of 5, 10 or 20 seconds per move, randomly sampled for each game. In S15, we show that this manipulation is effective at changing participants' response time. We predict that, if planning depth is a rough measure of the amount of computations a participant performs while making a move, it should scale with time used for that move[46, 4, 47]. Fig 3F shows that planning depth is overall lower than the learning experiment, and that planning depth indeed increases with longer time limits ($\beta = 0.042 \pm 0.018, p = 0.019$).

Despite this increase, however, we find no improvement in participant's playing strength ($\beta = -2.0 \pm 1.6, p = 0.21$, Fig 3E). The model suggests a potential explanation for the lack of performance: at the most relaxed time limit, people overlook features more often ($\beta = 0.0027 \pm 0.0010, p = 0.009$, Fig 3G), and the dropped features cancel out the benefit of increased search. Finally, in this experiment the heuristic quality does not change with time pressure ($\beta = 0.00086 \pm 0.00056, p = 0.13$, Fig 3H).

Generalization to large-scale mobile data. In all these experiments, we investigated expertise in participants recruited to perform a psychology experiment in a laboratory context.

It is not clear whether our expertise results will generalize to a more natural context for acquiring expertise. To address this issue, we collaborated with Peak, a mobile app company (`https://www.peak.net`), to collect a large-scale data set of users playing a visually enriched version (section S1.8) of 4-in-a-row at their leisure in their daily environment.

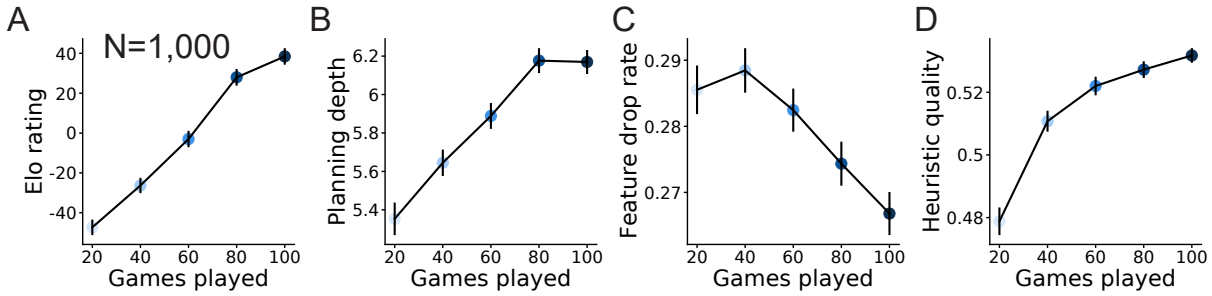

Figure 4: **The effects of expertise on planning in mobile data. A.** Average Elo rating of users of the mobile app, as a function of number of games played. In this panel and all others, error bars denote mean and standard error across participants. **B.** Average depth to which users plan, as estimated by the behavioral model. **C.** Same as **B**, for feature drop rate. **D.** Same **B**, for heuristic quality.

We analyze data from $1{,}000$ randomly selected users who played at least 100 games; this approximately matches the total experience of participants in our learning experiment. For each user, we grouped their experience into 5 blocks of 20 games, and estimated model parameters for each block (section S1.8). As before, playing strength ($\beta = 1.13 \pm 0.04, p < 0.001$, Fig 4A) and the depth of planning ($\beta = 0.0108 \pm 0.0010, p < 0.001$, Fig 4B) increase with experience, while the feature drop rate decreases ($\beta = -2.58 \cdot 10^{-4} \pm 4.7 \cdot 10^{-5}, p < 0.001$, Fig 4C). Once again, we validate that the increase in users' planning depth is not a result of slower play (section S18), replicating the results from the laboratory experiment. In this experiment, we also observe a reliable increase in heuristic quality ($6.12 \cdot 10^{-4} \pm 4.2 \cdot 10^{-5}, p < 0.001$, Fig 4D). However, the heuristic quality in the first 20 games of the mobile app data is much lower than that in the first session of the laboratory data (0.5301 ± 0.0098 vs $0.4788 \pm 0.0044, t(999) = 4.7, p < 0.001$). Therefore, the users have more opportunity to improve their feature weights, whereas heuristic

quality in the laboratory data might already start at ceiling.

Increased planning depth as pattern recognition. Previous chess literature has framed the superior performance of experts in terms of pattern recognition[9], often operationally defined through reconstruction experiments[2, 8]. We conducted a memory and reconstruction experiment with participants in the expertise experiment (See section S16), which shows that experts are better at reconstructing specifically those feature that our model relies on for evaluations. Thus, this data suggests a mechanistic explanation for the observed effect of expertise on planning depth. With expertise, players sharpen their representation of game-relevant features, allowing players to evaluate more positions per unit time and therefore plan more deeply. Hence, our results are consistent with improved pattern recognition in experts, but highlight the under-appreciated role of processing speed.

Discussion. In this article, we introduced a two-player combinatorial game of intermediate complexity that provides rich behavior, but for which cognitive modeling is still tractable. We demonstrated that a computational model based on a heuristic value function and forward search algorithm predicts human choices as well as response times and eye movements. Using this task and model, we showed robust evidence for increased planning and improved attention with expertise in both laboratory and large-scale mobile data.

Our fitting results indicate that our model best matches individual participants' choices with a planning depth of 4 to 6. However, this contradicts participants' anecdotal responses as well as the lower planning depths found in previous studies[22, 48].

First, we note that the planning depth result does not imply that people's plan is equally concrete for each of their next 4-6 moves. Our model contains value noise that is added along the nodes on the principal branch of the decision tree. Therefore, the model forms a concrete plan only for the first few moves, and later moves are planned more loosely.

Additionally, the model contains a sophisticated algorithm for deciding which nodes in the

tree to explore, but its decision as to when to terminate this search is random. In practice, this leads the model to often continue search without changing its eventual decision. By contrast, people have been shown to make such decisions more strategically and close to optimally[38]. In section S10, we show that amending the model with a threshold-based termination rule decreases the estimated planning depth, but the threshold cannot be identified from choice data. However, for any value of the threshold, we find a correlation between expertise and planning depth.

Would our results on the nature of expertise generalize to more complex games or natural planning tasks? We speculate that in more complex games, the same effects of expertise on attention and search will exist. For the heuristic quality effect, we note that in the laboratory data, participants already start with approximately correct inductive biases[49] about the relevant features and their relative values, and we observe no increase in heuristic quality with expertise. In the mobile app data, people's feature weights are initially worse and we do observe an increase. Thus, the model reveals a difference between laboratory and mobile data not obvious from playing strength alone.

Complex games like chess or Go contain many non-obvious features, which people can only learn through extensive experience or explicit instruction[50]. Therefore, we speculate that in such games, the superior performance of experts relies much more on domain-specific knowledge. Additionally, we note that 4-in-a-row, chess and Go are all deterministic two-player games, and expertise in planning with stochastic environments[51] or multi-agent interaction[52] might involve other computational mechanisms.

Since our model is framed generically, it can be adapted straightforwardly to study expertise in more complex planning tasks (such as chess, go, shogi or Connect-Four). However, it does require a set of features to define the heuristic function. The 4-in-a-row game contains a small set of simple features, which allow the model to explain people's choices. We discovered these

features through manual exploration and model comparison. For more complex games which require more sophisticated features, learning these features might prove more difficult. One promising approach would be to train neural networks to either play these games or predict human choices, and examine its internal representations. This approach could also open up a space of alternative models for 4-in-a-row.

Our modeling results show how experts differ from novice players, but do not shed light on how those differences emerge from their experience. In future work, we aim to analyze games from all 1.2 million users in the mobile data set, and match people's experience with their future actions. A promising candidate for modeling this learning process is deep reinforcement learning, specifically algorithms such as AlphaZero[53] and SAVE[54], which combine prior experience with forward planning at decision time.

Our work opens the door to a precise understanding of human planning across development and in patient populations. Additionally, it raises the question of how the components of the model are represented neurally. A specific hypothesis is that the value of future states is correlated with the activity of neurons associated with reward-based decision-making, such as those in orbitofrontal cortex[55]. Additionally, we predict that the time course of neural activity while a player contemplates their move reflects the dynamics of the value of the root node over iterations of the search algorithm.

References

[1] de Groot, A. D. *Het denken van den schaker: een experimenteel-psychologische studie* (Noord-Hollandsche Uitgevers Maatschappij, 1946).

[2] Chase, W. G. & Simon, H. A. Perception in chess. *Cognitive psychology* **4**, 55–81 (1973).

[3] Campitelli, G. & Gobet, F. Adaptive expert decision making: Skilled chess players search more and deeper (2004).

[4] Van Harreveld, F., Wagenmakers, E.-J. & Van Der Maas, H. L. The effects of time pressure on chess skill: an investigation into fast and slow processes underlying expert performance. *Psychological Research* **71**, 591–597 (2007).

[5] Sheridan, H. & Reingold, E. M. Chess players' eye movements reveal rapid recognition of complex visual patterns: Evidence from a chess-related visual search task. *Journal of vision* **17**, 4–4 (2017).

[6] Gobet, F. & Simon, H. A. Expert chess memory: Revisiting the chunking hypothesis. *Memory* **6**, 225–255 (1998).

[7] Bilalić, M., Langner, R., Erb, M. & Grodd, W. Mechanisms and neural basis of object and pattern recognition: a study with chess experts. *Journal of Experimental Psychology: General* **139**, 728 (2010).

[8] Linhares, A., Freitas, A. E. T., Mendes, A. & Silva, J. S. Entanglement of perception and reasoning in the combinatorial game of chess: Differential errors of strategic reconstruction. *Cognitive Systems Research* **13**, 72–86 (2012).

[9] Charness, N. Expertise in chess: The balance between knowledge and search. *Toward a general theory of expertise: Prospects and limits* 39–63 (1991).

[10] Saariluoma, P. Visuospatial and articulatory interference in chess players' information intake. *Applied Cognitive Psychology* **6**, 77–89 (1992).

[11] Holding, D. H. *The psychology of chess skill* (Lawrence Erlbaum, 1985).

[12] Holding, D. H. Evaluation factors in human tree search. *The American Journal of Psychology* **102**, 103–108 (1989).

[13] Gobet, F. & Jansen, P. Towards a chess program based on a model of human memory. *Advances in computer chess* **7**, 35–60 (1994).

[14] Gobet, F. A pattern-recognition theory of search in expert problem solving. *Thinking & Reasoning* **3**, 291–313 (1997).

[15] Holding, D. H. Counting backward during chess move choice. *Bulletin of the Psychonomic Society* **27**, 421–424 (1989).

[16] Holding, D. H. Theories of chess skill. *Psychological Research* **54**, 10–16 (1992).

[17] Charness, N. Expertise in chess and bridge. In *Complex information processing*, 203–228 (Psychology Press, 2013).

[18] Miller, K. J. & Venditto, S. J. C. Multi-step planning in the brain. *Current Opinion in Behavioral Sciences* **38**, 29–39 (2020).

[19] Daw, N. D., Niv, Y. & Dayan, P. Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature neuroscience* **8**, 1704 (2005).

[20] Huys, Q. J. *et al.* Bonsai trees in your head: how the pavlovian system sculpts goal-directed choices by pruning decision trees. *PLoS computational biology* **8**, e1002410 (2012).

[21] Huys, Q. J. *et al.* Interplay of approximate planning strategies. *Proceedings of the National Academy of Sciences* **112**, 3098–3103 (2015).

[22] Snider, J., Lee, D., Poizner, H. & Gepshtein, S. Prospective optimization with limited resources. *PLoS computational biology* **11**, e1004501 (2015).

[23] Kolling, N., Scholl, J., Chekroud, A., Trier, H. A. & Rushworth, M. F. Prospection, perseverance, and insight in sequential behavior. *Neuron* **99**, 1069–1082 (2018).

[24] Arad, A. & Rubinstein, A. The 11–20 money request game: a level-k reasoning study. *The American Economic Review* **102**, 3561–3573 (2012).

[25] Camerer, C. F., Ho, T.-H. & Chong, J.-K. A cognitive hierarchy model of games. *The Quarterly Journal of Economics* **119**, 861–898 (2004).

[26] Pfeiffer, B. E. & Foster, D. J. Hippocampal place-cell sequences depict future paths to remembered goals. *Nature* **497**, 74 (2013).

[27] Redish, A. D. Vicarious trial and error. *Nature Reviews Neuroscience* **17**, 147 (2016).

[28] Pezzulo, G., Donnarumma, F., Maisto, D. & Stoianov, I. Planning at decision time and in the background during spatial navigation. *Current Opinion in Behavioral Sciences* **29**, 69–76 (2019).

[29] Miller, K. J., Botvinick, M. M. & Brody, C. D. Dorsal hippocampus contributes to model-based planning. *Nature neuroscience* **20**, 1269 (2017).

[30] Groman, S. M., Rich, K. M., Smith, N. J., Lee, D. & Taylor, J. R. Chronic exposure to methamphetamine disrupts reinforcement-based decision making in rats. *Neuropsychopharmacology* **43**, 770–780 (2018).

[31] Akam, T. *et al.* The anterior cingulate cortex predicts future states to mediate model-based action selection. *Neuron* (2020).

[32] Beck, J. *Combinatorial games: tic-tac-toe theory*, vol. 114 (Cambridge University Press, 2008).

[33] Ma, W.-J. & van Opheusden, B. Tasks for aligning human and machine planning (2019).

[34] Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction*, vol. 1 (MIT press Cambridge, 1998).

[35] Bonet, B. & Geffner, H. Planning as heuristic search. *Artificial Intelligence. 2001 Jun; 129 (1-2): 5-33.* (2001).

[36] Dechter, R. & Pearl, J. Generalized best-first search strategies and the optimality of a. *Journal of the ACM (JACM)* **32**, 505–536 (1985).

[37] Griffiths, T. L., Lieder, F. & Goodman, N. D. Rational use of cognitive resources: Levels of analysis between the computational and the algorithmic. *Topics in Cognitive Science* **7**, 217–229 (2015).

[38] Callaway, F. *et al.* A resource-rational analysis of human planning. In *CogSci* (2018).

[39] Lieder, F. & Griffiths, T. L. Resource-rational analysis: understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences* 1–60 (2020).

[40] Lewis, R. L., Howes, A. & Singh, S. Computational rationality: Linking mechanism and behavior through bounded utility maximization. *Topics in Cognitive Science* **6**, 279–311 (2014).

[41] Treisman, A. M. & Gelade, G. A feature-integration theory of attention. *Cognitive psychology* **12**, 97–136 (1980).

[42] van Opheusden, B., Acerbi, L. & Ma, W. J. Unbiased and efficient log-likelihood estimation with inverse binomial sampling. *PLOS Computational Biology* **16**, e1008483 (2020).

[43] Acerbi, L. & Ji, W. Practical bayesian optimization for model fitting with bayesian adaptive direct search. In *Advances in Neural Information Processing Systems*, 1836–1846 (2017).

[44] Turing, A. Computing machinery and intelligence. *Mind* **59**, 433 (1950).

[45] Elo, A. E. *The rating of chessplayers, past and present* (Arco Pub., 1978).

[46] Chabris, C. F. & Hearst, E. S. Visualization, pattern recognition, and forward search: Effects of playing speed and sight of the position on grandmaster chess errors. *Cognitive Science* **27**, 637–648 (2003).

[47] Calderwood, R., Klein, G. A. & Crandall, B. W. Time pressure, skill, and move quality in chess. *The American journal of psychology* 481–493 (1988).

[48] Krusche, M. J., Schulz, E., Guez, A. & Speekenbrink, M. Adaptive planning in human search. *BioRxiv* 268938 (2018).

[49] Dubey, R., Agrawal, P., Pathak, D., Griffiths, T. L. & Efros, A. A. Investigating human priors for playing video games. *arXiv preprint arXiv:1802.10217* (2018).

[50] Charness, N., Tuffiash, M., Krampe, R., Reingold, E. & Vasyukova, E. The role of deliberate practice in chess expertise. *Applied Cognitive Psychology* **19**, 151–165 (2005).

[51] Brown, N. & Sandholm, T. Superhuman ai for multiplayer poker. *Science* **365**, 885–890 (2019).

[52] Anthony, T. *et al.* Learning to play no-press diplomacy with best response policy iteration. *arXiv preprint arXiv:2006.04635* (2020).

[53] Silver, D. *et al.* A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science* **362**, 1140–1144 (2018).

[54] Hamrick, J. B. *et al.* Combining q-learning and search with amortized value estimates. *arXiv preprint arXiv:1912.02807* (2019).

[55] Padoa-Schioppa, C. & Assad, J. A. Neurons in the orbitofrontal cortex encode economic value. *Nature* **441**, 223–226 (2006).

[56] Cornelissen, F. W., Peters, E. M. & Palmer, J. The eyelink toolbox: eye tracking with matlab and the psychophysics toolbox. *Behavior Research Methods, Instruments, & Computers* **34**, 613–617 (2002).

[57] Hunter, D. R. Mm algorithms for generalized bradley-terry models. *Annals of Statistics* 384–406 (2004).

[58] Sutton, R. S., McAllester, D. A., Singh, S. P. & Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, 1057–1063 (2000).

[59] de Groot, M. H. Unbiased sequential estimation for binomial populations. *The Annals of Mathematical Statistics* 80–101 (1959).

[60] Huyer, W. & Neumaier, A. Global optimization by multilevel coordinate search. *Journal of Global Optimization* **14**, 331–355 (1999).

[61] de Groot, A. D. *Het Denken van den sckaken* (Noord-Holland. Uitgev. Maatschappij, 1946).

1 Methods

We conducted six laboratory experiments: human-vs-human ($N = 40$ participants), generalization ($N = 40$), eye tracking ($N = 10$), learning ($N = 30$), time pressure ($N = 30$) and a Turing test ($N = 30$). The experiments can be played online on our website `basvanopheusden.github.com` (except for human-vs-human and eye tracking).

1.1 Participants

We recruited participants through the NYU Psychology research participant system, flyers, a sign-up link on our lab webpage or personal communication. We compensated participants $\$12$ per hour, but did not incentivize task performance. Participants provided informed consent and our experiments were approved by the Institutional Review Board of New York University.

1.2 Human-vs-human

For our human-vs-human experiment, we recruited 40 participants in pairs. For each pair, we provided consent forms and instructed participants on the task together, after which we separated them into different rooms. We manually started the task for each participant, then they played games against each other through an online interface. After 50 minutes had expired and they finished their last game, we manually proceeded them to a post-task questionnaire, during which we provided them with payment ($\$12$ in cash). Only after completing the survey and receiving payment did participants leave their respective rooms. Thus, participants interacted socially before and after the experiment, but not during the task.

Participants played games against each other, switching colors after every game. After each game, we presented both participants with a pop-up showing both players' names, current score and a button to continue to the next game. The interface proceeded only after both players had clicked the "continue" button. Every time the participant or their opponent moved, the interface

made a faint "clicking" noise. During games, instead of making a move, participants could offer a draw to their opponent, which caused a pop-up prompt to appear on the other participants' screen to accept or reject the offer. If the opponent accepted the draw, the game ended immediately, otherwise the pop-up disappeared and the player who made the offer could make a move instead. We did not restrict how many draw offers participants could make (including multiple draw offers on the same move), but participants made relatively few draw offers. In this experiment, we never imposed any time limits.

1.3 Generalization

In this experiment and all following ones, participants performed the task individually. Each session started with the participant providing informed consent, after which we instructed them on the details of the task. We always compensated participants $\$12$ at the end of their session.

In the generalization experiment, participants played against computer opponents for 30 minutes, after which they completed 82 trials each of a two-alternative forced-choice (2AFC) between moves in given board positions, and 82 board evaluation trials, in which they rated their winning chances in given board positions on a 7-point scale. Afterwards, we debriefed participants and provided payment. The interface for the play-against-computer task was identical to the human-vs-human experiment, except for two modifications: the between-game pop-up did not display any names or current score, and we removed the "offer draw" button.

In all human-vs-computer games, the computer's algorithm is similar to the behavioral model (see section 2.3) with 3 modifications: we used the pruning rule from the **Fixed branching** model, we included scale factors for weights of features belonging to the opponent (as in the **Opponent scaling** model) and for features of different orientation (the **Orientation-dependent weights** model) but not between "active" and "passive" feature weights (as in the No active scaling model). Finally, the algorithm used a slightly different feature set. We artifi-

cially added a "thinking time" to each computer move, which monotonically increased with the number of search iterations that the computer performed on each move. This ensured that the computer played faster in easy positions than in harder ones.

We created 30 computer opponents, all using the same algorithm but with different parameters. We started by fitting the behavioral model on participants in the human-vs-human experiment. For each parameter vector for a human participant, we created additional vectors by either dividing the feature drop rate by 2, doubling the mean tree size, halving the value noise or any combination thereof. We then ran an all-vs-all tournament between agents using these parameter vectors, and ranked their performance using the Elo system (2.1). Finally, we selected 30 agents such that their Elo ratings uniformly cover an interval ranging from slightly weaker than the worst human players to slightly stronger than the best. We divided the set of 30 agents into 6 levels with 5 agents per level, and matched participants with computer opponents using a one-up, one-down staircase, starting at level 3. For each game, we randomly selected an opponent from the 5 agents on the current level.

On a 2AFC trial, we presented a participant with a board position and two candidate options, and they indicated their preference by clicking on the corresponding candidate move (Fig S1A). We did not impose any time limits on participants' choices. To present participants with interesting choices and to ensure that participants' choices constrain model parameters, we selected board positions that maximize mutual information between the chosen move and model parameters, within the set of parameter vectors inferred by the model for human participants. Additionally, we computed the objective value of each move, and ensured that each trial type (both moves winning, one winning and one drawn, both drawn, etc) is represented equally (14 times). We presented the same positions to each participant, in shuffled order.

In the evaluation experiment, we presented participants with pre-arranged board positions and instructed them to indicate their expected winning chances on a 7-point scale (see Fig S1B).

Participants entered their rating by clicking one of 7 buttons. We labeled the first, middle and last button with “losing”, “equal” and “winning”, respectively. For the evaluation experiment, we selected positions using the same procedure as in the 2AFC experiment, except that in the final selection stage, we ensured that the game-theoretic values (S2.4) of the presented positions were equally distributed across winning, losing or drawn (28 each).

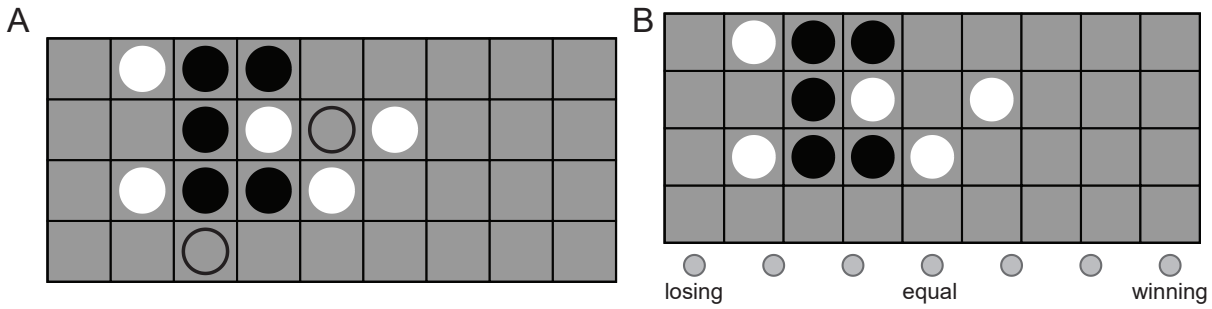

Figure 1: **A.** Example board in the 2AFC task. The participant, playing black, chooses between one of two candidate moves in a given board position. **B.** Example evaluation board. The participant ranks the winning chances of the current player on a 7-point scale.

1.4 Turing test

The Turing test experiment consisted of two sessions on consecutive days. On the first session, participants played against computer opponents for 60 minutes. In this experiment, each computer opponent followed the main model with parameters inferred for an individual participant in the human-vs-human experiment.

On the second session, participants performed 180 trials of a classification task (see Fig 15). On each trial, we presented participants with a movie of a segment of a game played either by two players in the human-vs-human experiment, or two computers following the main model with parameters inferred for those players. Participants could start the movie at any time by pressing a “Play” button, and the video played at a constant speed of 1.8 seconds per move. After the video, participants judged the video using a slider labeled “Certainly computers” on the

left, “No clue” in the middle and “Certainly humans” on the right. After each trial, we provided participants with feedback whether their classification judgment was correct (“Correct!”) or not (“Incorrect.”).

We selected game segments to use for human-vs-human videos from games played in human-vs-human experiment, and computer-vs-computer videos using a similar sampling method. First, we created one video excerpt from each game in the human-vs-human experiment. For each game, we drew a number from a geometric distribution with rate 0.15 and selected the position that occurred in the game after that many moves. We then drew a maximum length for the segment from another geometric distribution with rate 0.1, and added moves from the game until the segment exceeded that maximum length or until the end of the game. For each game, we also generated a computer-vs-computer segment, starting from the same position, using the same maximum length. As before, we added moves from a simulated computer-vs-computer game until that segment exceeded the maximum length or the game ended. In other words, all computer-vs-computer video segments start from a position that occurred in a human-vs-human game, but all moves are made by the behavioral model. Because of this sampling method, and the constant playback speed of the videos, the only cues available to participants are the moves played and not the starting position or response times. Finally, we selected a random subset of 90 games to use for human-vs-human videos and 90 others for computer-vs-computer videos.

To instruct participants on the task, we used the following text: “*Today, you will be shown 180 short videos, either from games between two human players or between two computers. Half of the videos are from games between humans, the other half between computers. The videos may start from any point in a game, so the starting position is not necessarily an empty board. Your task is to identify if the video is from a human-vs-human game or a computer-vs-computer game. You will also be asked to report how confident you are about your choice. There is no time limit to this task.*”

1.5 Eye tracking

In the eye tracking experiment, participants play against computer opponents for 40 minutes and performed 82 trials of the 2AFC experiment, with settings for both experiments identical to the generalization experiment above. For the entire experiment, we recorded their eye movements with a remote infrared video-oculographic system (EyeLink 1000; SR Research, Ltd., Mississauga, Ontario, Canada[56]) with a 1 kHz sampling rate and ≈ 0.01 degree precision. We acquired eye position data with the EyeLink software using the "Heuristic filter ON" option. We displayed stimuli on a 21-inch Sony GDMF520 CRT monitor (resolution: 1280×960 pixels, refresh rate: 100 Hz). Subjects used a headrest located approximately 57 cm from the screen. We set the eye tracker to record events only, so that our data set consists of a time series of fixations, saccades and blinks.

On each session, we first calibrated the eye tracker with the built-in 9-point calibration method, but we also added a calibration condition directly before and after the play-against-computer component of our experiment. In this calibration procedure, we presented an empty board with a white piece with a fixation cross on top of it on the bottom left tile (Fig S2A). We instructed participants to fixate on the cross and press the space bar when they felt their fixation was steady. After they pressed the space bar, the piece and the cross moved one tile to the right, instructing the participant to fixate on the next tile, which they again indicated with a space bar press. We continued moving the cross accordingly across all 36 squares, obtaining fixation coordinates and time stamps for the space bar presses for each square.

1.6 Learning

The learning experiment consisted of 5 sessions. We required participants to schedule consecutive sessions with no more than 2 days in between. On the first, third and fifth session, participants played against computer opponents for 30 minutes and completed 60 trials each

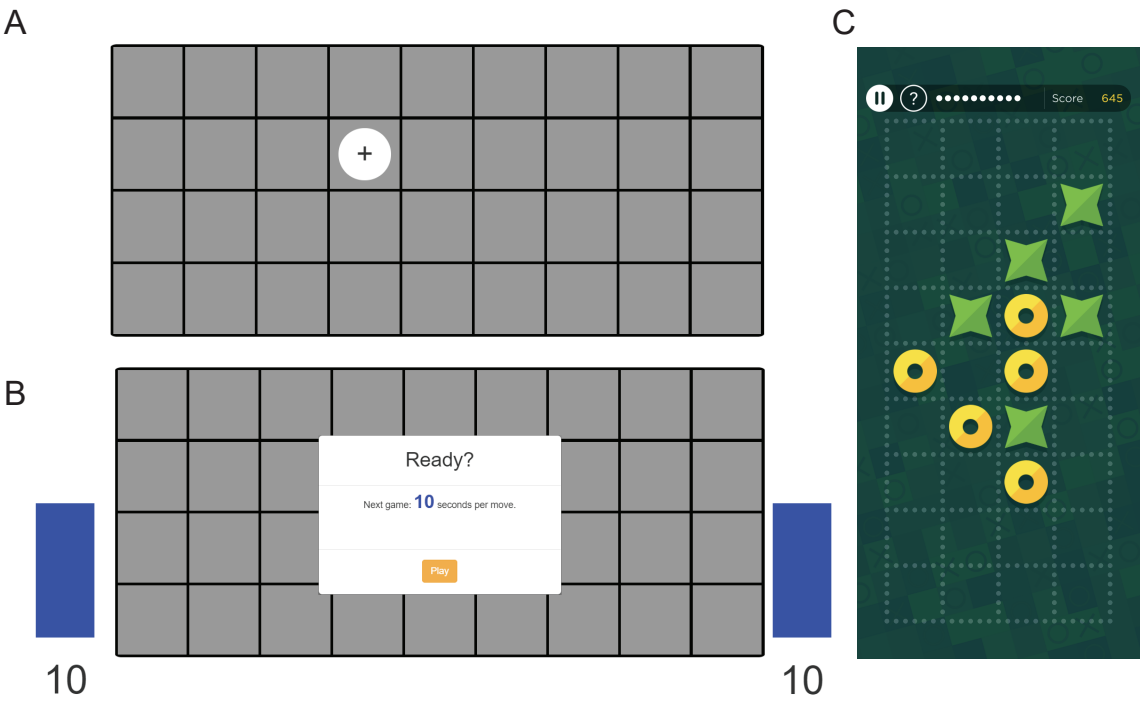

Figure 2: **A.** Screenshot of the calibration component of the eye tracking experiment. We instructed participants fixate on the black cross in the white circle, and press the space bar. The white circle and the cross then move to the next tile, until we have acquired fixation coordinates for all 36 tiles. **B.** Screenshot of the pre-game pop-up in the time pressure experiment. The time limit is indicated on the pop-up, and underneath the visual timer on the right of the board. Additionally, the height and color of the colorbar inform the participant of the time limit. During each participant's move, the colorbar on the right gradually shrinks while the text beneath counts down to zero. The same happens for the left timer during each computer's move. **C**. Screenshot of the game interface on the mobile application.

of the 2AFC and evaluation conditions. On the second and fourth sessions, participants played against computer opponents for the entire 60-minute session. In all these sessions, the computer opponents were identical to those of the generalization experiment and the positions were selected using the same information criteria, with one difference. We selected 180 positions that we divided into 3 groups of 60, and ensured that the order of the days on which we presented these positions was counterbalanced across participants. We compensated participants $\$12$ per session, with a $\$12$ completion bonus at the end.

1.7 Time pressure

In the time pressure experiment, participants played against computer opponents for 50 minutes, again with the identical procedure as the generalization experiment. However, in each game, both the human participant and the computer opponent had to obey a time limit of 5, 10 or 20 seconds per move. The time constraint was constant within each game and varied randomly between games. If a participant exceeded this time limit, the game ended immediately and counted as a loss. We also amended the "thinking time" for the computer, to ensure that it never uses more than 80% of its allotted time. However, we emphasize that this did not change the computer opponent's decisions, as those take only a fraction of a second to compute. We control the opponent's thinking time by simply pausing the interface for the appropriate amount of time.

To inform participants of the time constraint, we indicated the time limit for each game in a pop-up before the start of that game. Additionally, directly to the right of the board we displayed a timer; a colored bar that shrunk gradually while participants were contemplating their move. Directly below the timer, we displayed a text-based count-down with the remaining thinking time in seconds (Fig S2B). In the 20-second condition, at the start of each move the colorbar was equally high as the board and linearly decreased to zero in 20 seconds. Initially,

the bar was green, but when the participant had 10 seconds left it changed color to blue, and 5 seconds before the end it changed color to red. To warn participants even more of the passage of time, we played three warning sounds (short beeps) when the participant had 2, 1 or 0 seconds left, with increasingly higher pitch as time counted down. In the 5 or 10-second condition, we started the timer in the same state it would be in the 20-second condition after 10 or 15 seconds had elapsed (10 seconds: blue colorbar, half as high as the board; 5 seconds: red bar, quarter board height). When the computer was "thinking", we displayed a timer to the left of the board, with the identical behavior. The time warnings were largely effective, and participants lost on time in only 1.87% (33 out of 1766) of their games.

1.8 Large-scale mobile data

In collaboration with the mobile app company Peak (`https://www.peak.net`), we collected a data set of people playing 4-in-a-row. When signing up for the app, users consented to a privacy policy, which included a provision that aggregated and anonymized data might be shared with third parties such as universities. The Institutional Review Board of New York University determined that no further consent was required and approved the research protocol as "exempt".

We collected 10,874,547 games from 1,234,844 unique users. Users always play first, and the game board itself is vertically-oriented and gamified (Fig S2C). Additionally, users play at-will against a computer opponent implementing a version of our main model, with parameters adapted from fits on data collected in the laboratory experiments (human-vs-human, generalization, eye tracking, learning, and time pressure). The procedure for generating the computer opponents is identical to the one in the generalization, learning, and time pressure experiments, but we re-calibrated the computer opponents since they always play second. We created 7 classes of computer opponents of varying strength, and matched users with an opponent based

on their track record of game results. For analysis, we randomly selected $1,000$ participants from this data set that had each played at least 100 games. We grouped their experience into 5 blocks of 20 games in order to approximately match the total experience level of participants in the learning experiment.

1.9 Memory and reconstruction experiment

In this experiment, participants memorized and reconstructed board positions. We recruited two groups of 19 participants. The first group consisted of participants who had previously completed the learning experiment, no more than 4 weeks prior. The second group had no prior experience with the game and were informed that the task involved memorizing patterns of squares and circles.

On each trial, we presented participants with board positions for 10 seconds followed by a blank board for 1 second. We then prompted them to reconstruct the original position without a time limit. The reconstruction interface allowed participants to right-click on any square to place or remove a black piece and to left-click to place or remove a white piece. At any time, participants could click a "submit" button to indicate they had finished their reconstruction, after which they received feedback indicating the fraction of the 36 squares correctly reconstructed, including empty squares.

Each participant reconstructed the same set of 96 positions in a random order, in an approximately one-hour session. We generated two sets of 48 positions. The first set contained positions from human-vs-human games. To generate this set, we varied the number of pieces in each position from 11 to 18, and randomly selected 6 positions from human-vs-human games with that number of pieces. The second set consisted of procedurally generated positions, constrained to exactly match the distribution of the number of pieces, and approximately match the marginal distribution of occupied squares.

2 Analysis methods

2.1 Playing strength estimation using Bayeselo

To estimate a player's playing strength from games against computer opponents, we use Elo ratings[45], implemented using the publicly available program `Bayeselo`[57]. To measure Elo ratings of all players in all experiments against a common baseline, we run `Bayeselo` on a database containing all human-vs-computer games and a simulated computer-vs-computer tournament, in which each computer plays once against every other computer, including itself. In the computer-vs-computer tournament, we include both the agents used in the generalization, learning and time pressure experiment as well as the agents used in the mobile app.

2.2 Model specification

We assume that people's choices on each move are independent and generated by the same decision-making process with the same parameters within a single session. We first describe the model broadly, then in more detail. Our model is based on heuristic search [34], and consists of a value function and a tree search algorithm. Additionally, we include sources of noise to capture variability in human play and human-like mistakes.

Value function. The core of our model is a value function $V(s, \vec{w})$, which assigns a value to a board state s. The higher this value, the more likely the black player is to win from that state. We assume that people use value function approximation[58], and that people's value function is a weighted sum of features

$$V(s, \vec{w}) = \sum_{i=1}^{5} w_i \phi_i(s, \text{self}) - \sum_{i=1}^{5} w_i \phi_i(s, \text{opponent}) \tag{1}$$

where ϕ_i denote the features and w_i the weights. In the following, and in the main text, we omit the dependence of $V(s, \vec{w})$ on $\vec{w}$ for brevity. The value function uses 5 features: center, connected 2-in-a-row, unconnected 2-in-a-row, 3-in-a-row and 4-in-a-row. The center feature

assigns higher value to squares near the center of the board. The other features count how often their corresponding patterns occur on the board (horizontally, vertically, or diagonally). Whenever the model evaluates a state, the weights of features belonging to the active player are multiplied by a scaling constant C. For example, the three-in-a-row feature is more valuable when it's the player's own move (it's an immediate win) than on the opponent's (it can be blocked). Active scaling does not apply to the center feature.

Tree search. The evaluation function guides the construction of a decision tree with an iterative best-first search algorithm[36]. Each iteration, the algorithm chooses a board position to explore further, evaluates the positions resulting from each legal move, and prunes all moves with value below that of the best move minus a threshold. After each iteration, the algorithm stops with a probability γ, resulting in a geometric distribution over the total number of iterations.

Noise. To account for variability in people's choices, we add three sources of noise. We model selective attention by randomly dropping features (at specific locations and orientations) before constructing the decision tree, which are then omitted during the calculation of $V(s)$ anywhere in the tree. During tree search, we add Gaussian noise to $V(s)$ in each node. Finally, we include a lapse rate λ.

2.3 Detailed model specification

Value function. The value function consists of two terms, the first of which measures whose pieces are closer to the board center:

$$V_{\text{center}}(s) = \sum_{\vec{x} \in \text{Pieces(s,black)}} \frac{1}{\|\vec{x} - \vec{x}_{\text{center}}\|} - \sum_{\vec{x} \in \text{Pieces(s,white)}} \frac{1}{\|\vec{x} - \vec{x}_{\text{center}}\|} \tag{2}$$

where $\text{Pieces}(s, p)$ enumerates the locations of all pieces that player p owns, $\vec{x}_{\text{center}}$ denotes the coordinate of the board center, and $\|\cdot\|$ is the Euclidean distance.

The second term counts how often particular patterns occur on the board (horizontally, vertically, or diagonally). A feature is a binary function $f_{t,x,y,o}(s)$ that returns 1 if a pattern of type t occurs at location (x, y) with orientation o, and 0 otherwise. We use the following 4 patterns:

1 *Connected 2-in-a-row:* two adjacent pieces with enough empty squares around them to complete 4-in-a-row.

2 *Unconnected 2-in-a-row:* two non-adjacent pieces that lie on a line of four contiguous squares, with the remaining two squares empty.

3 *3-in-a-row:* three pieces that lie on a line of four contiguous squares, with the remaining square empty. This pattern represents an immediate winning threat.

4 *4-in-a-row:* four pieces in a row. This pattern appears only in board states where a player has already won the game.

We define F to be the set of all such features (one for each type, orientation and board location), and associate a weight w to each feature in this set. The feature weight depends only on its type, not the orientation or location. Finally, we write the value function as:

$$V_F(s) = w_{\text{center}} V_{\text{center}}(s) + c_{\text{black}} \sum_{i \in F} w_i f_i(s, \text{black}) - c_{\text{white}} \sum_{i \in F} w_i f_i(s, \text{white}) + \mathcal{N}(0, 1) \quad (3)$$

where $c_{\text{black}} = C$ and $c_{\text{white}} = 1$ whenever black is to move in state s, and $c_{\text{black}} = 1$ and $c_{\text{white}} = C$ when it is white's move. The scaling constant C captures value differences between "active" and "passive" features. For example, a black three-in-a-row feature signals an immediate win on the black's move, but not on white's. The last term $\mathcal{N}(0, 1)$ represents additive Gaussian noise with mean zero and unit variance.

Search algorithm. The search algorithm constructs a decision tree, consisting of nodes that contain a state s, the color of the active player in that state and a value associated to the state.

Upon initialization, the value of a new node is set by calling the feature-based evaluation function. However, this value changes as the algorithm investigates the consequences of future play from that state. The algorithm starts with a single-node decision tree and gradually grows the tree. Each iteration, the algorithm selects a leaf node, expands it by adding one child node each for a number of candidate moves, and backpropagates the value of these new nodes recursively into the leaf node as well as its parents.

Algorithm 1: MakeMove(State s)

```
if Lapse(λ) then
    return RandomMove(s);
else
    DropFeatures(δ);
    root ←node(s);
    while !Stop(γ) and !Determined(root) do
        n ←SelectNode();
        ExpandNode(n);
        Backpropagate(n);
    end
end
return argmax_{c∈children(root)} c.val;
```

Here, `Lapse` and `Stop` represent stochastic functions that return `true` with probability λ and γ, respectively, and `Determined` checks if the value of the root node (winning, losing or drawn) has been determined with certainty. `RandomMove`(s) returns a random legal move in state s. The `SelectNode` function determines the order by which nodes are added to the tree. We use best-first search, which selects a node by following the principal variation, in which both players always make the best moves according to the currently estimated values, starting from the root to a leaf node. Because the value of nodes in the tree change after each iteration, so does the principal variation, and therefore the search algorithm dynamically switches between different branches of the tree.

Algorithm 2: SelectNode()

$n \leftarrow$ root;
while *children*$(n) \neq \emptyset$ **do**
  if $n.color =$ *black* **then**
    $n = \arg\max_{c \in \text{children}(n)} c.val$
  else
    $n = \arg\min_{c \in \text{children}(n)} c.val$
return n;

After the search algorithm has selected a leaf node to explore, it expands it by adding one child node for each legal move in the associated state. As it initializes the children, it automatically evaluates their states using $V(s)$ as defined above. The algorithm does not yet check whether either of these states is terminal (that is, either player has achieved 4-in-a-row or the board is full), but it effectively does so if $w_{\text{4-in-a-row}}$ is high enough. Next, the algorithm prunes unpromising children; those whose value difference with the best candidate move exceeds a threshold θ. Only afterwards does it assign $V = 10,000$ to each child state in which black has won, $V = -10,000$ if white has won and $V = 0$ for draws. It is therefore possible that, if $w_{\text{4-in-a-row}}$ is too low, or if the algorithm has dropped a 4-in-a-row feature in a relevant location, it will prune away an immediately winning move, which can result in gross (but human-like) blunders.

Algorithm 3: ExpandNode(node n)

s$\leftarrow n.state$;
foreach *legal move* m *in* s **do**
  n.`AddChild(node(`$s + m$`));`
if $n.color =$ *black* **then**
  $V_{\text{max}} = \max_{c \in \text{children}(n)} c.val$
else
  $V_{\text{max}} = \min_{c \in \text{children}(n)} c.val$
for $c \in$ *children*(n) **do**
  if $|c.val - V_{max}| > \theta$ **then**
    `RemoveChild(`c`)`

Next, the search algorithm incorporates the value of the newly created nodes into the deci-

sion tree with minimax backpropagation. After backprogagation, the value of each state reflects the search algorithm's best estimate of the result of a game starting in that state with perfect play from both sides.

Algorithm 4: Backpropagate(node n)

if *n.color=black* **then**
  $n.val \leftarrow \max_{c \in \text{children}(n)} c.val$
else
  $n.val \leftarrow \min_{c \in \text{children}(n)} c.val$
if *n≠root* **then**
  `Backpropagate` (n.parent)

The search algorithm continues to run until the `Stop` routine returns `true`, after which it makes the best move according to its estimated values. Since the `Stop` routine is random and independently drawn each iteration, the total number of iterations follows a geometric distribution with parameter γ. When implementing our model as an AI algorithm to play against human opponents, we convert the number of iterations N into a 'thinking time' for the AI by $t = a\sqrt{N\gamma} + b$, where $a = 4$s and $b = 0.5$s.

The main model has 10 parameters: the pruning threshold θ, the stopping probability γ, the lapse rate λ, the feature drop rate δ, the active scaling constant C and the feature weights w_{center}, $w_{\text{connected 2-in-a-row}}$, $w_{\text{unconnected 2-in-a-row}}$, $w_{\text{3-in-a-row}}$, $w_{\text{4-in-a-row}}$. We do not add a parameter for the variance of the value noise, since changing the noise distribution from $\mathcal{N}(0,1)$ to $\mathcal{N}(0,\sigma^2)$ has the same effect as changing $\theta \to \frac{\theta}{\sigma}$ and $w \to \frac{w}{\sigma}$ for each feature. Therefore, adding σ would over-parametrize the model and by construction cause σ, θ and $\{w_i\}$ to be unidentifiable from data.

2.4 Computing game-theoretic values

We can use the model to calculate the game-theoretic value $\tilde{V}(s)$ of a position s, that is, the outcome of a game starting from position s with perfect play from both sides. To compute the game-theoretic value, we execute the best-first search algorithm with default feature weights,

no sources of noise and no pruning. In the limit of infinitely many iterations, the value of the root node in the decision tree of best-first search is guaranteed to converge to the game-theoretic value. In practice we found that $200{,}000$ search iterations was sufficient for almost all positions. For positions in which $200{,}000$ iterations did not yield a determined result, we set the game-theoretical value to $\tilde{V}(s) = 0$, in other words, a draw.

2.5 Alternative model specifications

2.5.1 Lesions

Our first set of alternative models are lesion models, obtained by removing components from the main model. Each lesion can be implemented by fixing a parameter to a constant. The **No center**, **No connected 2-in-a-row**, **No unconnected 2-in-a-row**, **No 3-in-a-row** and **No 4-in-a-row** models are obtained by setting the respective feature weight to zero. The **No feature drop** model is obtained by fixing δ to zero, and the **No active scaling** model results from fixing C to 1. To obtain the **No pruning** model, we fix θ to $20,000$, which is larger than any value difference that occurs in search and causes the model to never prune. Note that the model cannot compensate by increasing feature weights since their order of magnitude is yoked by fixing the value noise to have unit variance. Finally, the **No tree** model is achieved by fixing γ to 1. This causes the algorithm to stop after 1 iteration, in which case it will have expanded only the root node, and its choice will be the highest-value child. Pruning lower-value children does not affect this choice, so θ is not a parameter in this model.

2.5.2 Modifications

In our first modified model, **Fixed iterations**, we change the stopping routine `Stop` from a stochastic function to a deterministic function that returns `true` whenever the number of iterations has exceeded a constant N. In the **Fixed depth** model, we amend the search process to explore every branch up to a fixed depth D. In the **Fixed branching** model, we amend the

pruning rule to keep the K highest-value children in each node (lowest-value when white is to move). If the expanded node has less than K children, the algorithm prunes nothing. Next, we consider removing the feature drop mechanism and instead applying a function in which each child is pruned with a probability ε in `ExpandNode` before the value-based pruning, resulting in the **Tile dropping** model. For the **Optimal weights** model, we restrict the feature weights $\{w_i\}$ to a constant vector, which we chose by maximizing the Pearson correlation between $\tanh\left(V(s)/20\right)$ and the game-theoretic value $\tilde{V}(s)$ across all states s that occurred in the human-vs-human experiment (see S2.4).

Finally, we consider **Monte Carlo Tree Search** (MCTS). In this algorithm, instead of evaluating a state with $V(s)$, we perform a rollout; a simulated game starting from state s between two agents that follow a myopic policy. That is, in state s', the agent chooses the move m that maximizes $V(s'+m)$, or the one that minimizes it when white is to move. We then assign a value of 1 to state s if the rollout results in a win for black, 0 for white wins, and $\frac{1}{2}$ if the game is a draw. Note that, since the evaluation function contains noise, the myopic policy and the outcome of the rollout are also stochastic. Note also that we perform only a single rollout when evaluating a state.

After performing a rollout, MCTS backpropagates by averaging rather than minimax, ensuring that the value of each intermediate node of the tree is equal to the average outcome of the rollouts conducted in all descendants of that node. We also amend the best-first selection rule

$$n = \underset{c \in \text{children}(n)}{\text{argmax}}\ c.\text{val} \tag{4}$$

to the UCB formula

$$n = \underset{c \in \text{children}(n)}{\text{argmax}}\ c.\text{val} + C_{\text{exp}} * \sqrt{\frac{\log\left(n.N_{\text{rollouts}}\right)}{c.N_{\text{rollouts}}}} \tag{5}$$

where $n.N_{\text{rollouts}}$ counts the number of rollouts that have been conducted in node n or any of its descendants, and C_{exp} is a parameter that controls the balance between exploitation (investigat-

ing high-value children) and exploration (investigating children that haven't been investigated much). Finally, after the tree search terminates, the algorithm makes a move by maximizing N_{rollouts} across all children of the root node.

2.5.3 Extensions

We create the **Orientation-dependent weights** by multiplying the weight of vertically or diagonally oriented features by scaling constants c_{vert} and c_{diag}, respectively. For the **Orientation-dependent dropping** model, we allow the feature drop rate for horizontally, vertically or diagonally oriented features to vary, whereas in the **Type-dependent dropping**, we let the drop rate depend on the feature type. In the **Triangle** model, we include a feature which counts the number of times that any of a set of 3-piece patterns occurs on the board. Finally, the **Opponent scaling** model extends the main model by adding a scaling constant c_{opp} that multiplies weights of features belonging to the opponent. Note that opponent scaling and active scaling are dissociated since the former multiplies weights of the opponent's features regardless of whose move it is, whereas the latter is adaptive.

2.6 Model fitting

The main model has 10 parameters: the 5 feature weights, the active-passive scaling constant C, the pruning threshold θ, stopping probability γ, feature drop rate δ and the lapse rate λ. We infer these parameters for individual participants and individual learning sessions or time limit conditions with maximum-likelihood estimation. Unfortunately, deriving the log-likelihood analytically requires marginalization of all latent variables (which features are dropped, the value at each node and the number of iterations in the search algorithm), which is intractable. Restricting ourselves to only models with analytical likelihoods would limit the types of models that one can consider, particularly in regards to the noise structure. Instead, we estimate the

log-likelihood with inverse binomial sampling (IBS) [59, 42], a method that estimates the log-likelihood by comparing the data to simulated data generated from the model. IBS is unbiased but its estimates are noisy. Moreover, we cannot calculate gradients of the log-likelihood, so we optimize the log-likelihood with multilevel coordinate search [60], a gradient-free algorithm. To reduce overfitting, we compare models with 5-fold cross-validation.

The fitting pipeline is computationally expensive, and fitting one participant's data for a single model requires approximately 10^{14} floating point operations. We perform the model fits on the NYU high performance cluster (Intel Xeon E5-2690v2 CPUs 3.0GHz) with a parallel implementation of IBS, which uses 20 cores. On our hardware, fitting takes approximately 1 hour for one participant and one model.

In this article, we fit all 22 models on 330 data sets, and the main model on an additional $5{,}000$ data sets. The computational expense of model fitting implies that some common statistical analyses of model reliability like parameter and model recovery are intractable. Therefore, in section S4, we assess the reliability and robustness of our modeling methods using less computationally expensive methods.

2.7 Derived metrics

To analyze the nature of expertise and the effect of time pressure, we convert the set of 10 parameters from the main model to 3 derived metrics: planning depth, feature drop rate and heuristic quality.

We the define the planning depth as the length of the principal variation in the model's decision tree, averaged across simulations of the model with a given a parameter vector in a fixed set of probe positions, specifically, all positions that occurred in the human-vs-human experiment ($5{,}482$ positions). As in algorithm 2, the principal variation is the sequence in which both players make the best move according to the values in the decision tree, from the root to

a leaf. The length of this sequence is equal to the depth of that leaf node, and reflects how far into the future the model plans. We average this depth across 10 simulated moves, and across all probe positions. Note that, because the set of probe positions is fixed, the planning depth is only a function of the model parameters.

The feature drop rate is simply the parameter δ. To define the heuristic quality, we evaluate $V(s, \vec{w})$ in all the probe positions, and compute the Pearson correlation between $\tanh(V(s, \vec{w})/20)$ and the game-theoretic value $\tilde{V}(s)$ (see S2.4). Note that the heuristic quality only depends on the feature weights $\vec{w}$ and the active scaling constant C.

Because the probe positions are fixed in the definition of planning depth, it is purely a function of the model parameters. Planning depth depends primarily on the stopping probability (Spearman correlation: $\rho = -0.87, p < 0.001$), and there is a minor dependence on the pruning threshold ($\rho = -0.21, p < 0.001$). These correlations are computed across a range of parameter vectors taken from model fits to human data. The heuristic quality is a more complicated function of the feature weights and active scaling constant. For example, the heuristic quality correlates with $w_{\text{3-in-a-row}}/w_{\text{connected 2-in-a-row}}$, but the correlation is relatively weak ($\rho = 0.55, p < 0.001$), and other feature weights influence the heuristic quality too.

In other words, the derived metrics carve up the set of 10 parameters: planning depth primarily depends on pruning threshold and stopping probability, feature drop rate on the feature drop rate, and heuristic quality solely depends on feature weights. Together, the three metrics provide a reduced representation of the model parameters that is more interpretable, more reliably inferred (Fig S4) and sufficient to capture the increase in performance across sessions (Fig S20).

2.8 Data and Code Availability

Data and code that support the findings of this study are available through the Open Science Framework (`https://osf.io/n2xjm/`).

Acknowledgments

We thank Zeyan Shu for piloting an early version of the experiment and Feroz Khalidi for assistance with data collection. We thank Andra Mihali, Aspen Yoo, Maija Honig, Luigi Acerbi, Will Adler, Fred Callaway, Marcelo Mattar and other current members and alumni of the Ma Lab for helpful discussions. This work was supported by grant number IIS-1344256 to W.J.M. and by Graduate Research Fellowship number DGE1839302 to I.K. from the National Science Foundation.

Author contributions

All authors contributed to conceptualization of the research. BvO, GG, IK and YL collected data. BvO, GG, IK, YL and ZB developed software, methodology and performed analysis. BvO, IK and WJM wrote the paper. WJM supervised the project and acquired funding.

Competing interests

The authors declare no competing interests.

Additional Information

Supplementary Information is available for this paper.

Correspondence and requests for materials should be addressed to basvanopheusden@nyu.edu

Reprints and permissions information is available at www.nature.com/reprints.

Supplementary Information for

Revealing the impact of expertise on human planning with a two-player board game

Bas van Opheusden, Gianni Galbiati, Ionatan Kuperwajs, Zaha Bnaya, Yunqi Li, Wei Ji Ma

1 Statistics

To analyze changes in Elo rating and derived metrics in the learning and time pressure experiment, and in the mobile data, we perform mixed effects regressions with random intercepts for every participant. In each regression, we treat the condition (i.e., session number for the learning experiment, time limit for time pressure, and games played in the mobile data) as a continuous predictor. To assess whether the condition has a significant effect, we performed a likelihood ratio test between an unrestricted regression predicting depth with a fixed effect of condition, and a restricted regression with no fixed effect. We performed all regressions using the `lme4` library in Rstudio.

Fixed effects:	Estimate	Std. Error	T value
Intercept	54.753	21.319	2.568
Session number	21.593	4.631	4.663
Random effects:	Variance	Std.Dev.	
Participant	9775	98.87	

Table 1: Coefficients in a linear mixed-effects regression analysis predicting Elo rating in the learning experiment treating session number as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
With session number	1829.7	1838.8	-911.86	1823.7			
Without session number	1811.6	1823.6	-901.80	1803.6	20.137	1	< 0.001

Table 2: Likelihood ratio test between the model in table 1, and a restricted regression without the fixed effect of session number.

Fixed effects:	Estimate	Std. Error	T value
Intercept	5.23508	0.24443	21.418
Session number	0.25477	0.06056	4.207
Random effects:	Variance	Std.Dev.	
Participant	1.132	1.064	

Table 3: Coefficients in a linear mixed-effects regression analysis predicting planning depth in the learning experiment treating session number as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without session number	515.10	524.13	-254.55	509.10			
With session number	500.47	512.51	-246.23	492.47	16.637	1	< 0.001

Table 4: Likelihood ratio test between the model in table 3, and a restricted regression without the fixed effect of session number.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.221553	0.011568	19.153
Session number	-0.011932	0.002796	-4.267
Random effects:	Variance	Std.Dev.	
Participant	0.002607	0.05106	

Table 5: Coefficients in a linear mixed-effects regression analysis predicting feature drop rate in the learning experiment treating session number as a continuous predictor and controlling for participant-specific effects using random intercepts.

2 Estimating the number of non-terminal states

Any non-terminal state in the 4-in-a-row game needs to satisfy two conditions. First, the number of black pieces is either equal to the number of white pieces (when black is to move), or

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without session number	-405.11	-396.08	205.56	-411.11			
With session number	-420.20	-408.15	214.10	-428.20	17.084	1	< 0.001

Table 6: Likelihood ratio test between the model in table 5, and a restricted regression without the fixed effect of session number.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.535416	0.008634	62.01
Session number	-0.006702	0.002025	-3.31
Random effects:	Variance	Std.Dev.	
Participant	0.001499	0.03871	

Table 7: Coefficients in a linear mixed-effects regression analysis predicting heuristic quality in the learning experiment treating session number as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without session number	-506.13	-497.10	256.07	-512.13			
With session number	-514.70	-502.66	261.35	-522.70	10.569	1	0.00115

Table 8: Likelihood ratio test between the model in table 7, and a restricted regression without the fixed effect of session number.

Fixed effects:	Estimate	Std. Error	T value
Intercept	-42.017	30.823	-1.363
Time limit	-1.999	1.599	-1.250
Random effects:	Variance	Std.Dev.	
Participant	15080	122.80	

Table 9: Coefficients in a linear mixed-effects regression analysis predicting Elo rating in the time pressure experiment treating time limit as a continuous predictor and controlling for participant-specific effects using random intercepts.

one higher (when white is to move). Second, neither player has 4 pieces in a row, column or diagonal. Any state meeting these two conditions can be reached in a legal game. Therefore,

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without time limit	1133.9	1141.4	-563.95	1127.9			
With time limit	1134.3	1144.3	-563.17	1126.3	1.5682	1	0.2105

Table 10: Likelihood ratio test between the model in table 9, and a restricted regression without the fixed effect of time limit.

Fixed effects:	Estimate	Std. Error	T value
Intercept	3.60147	0.29462	12.224
Time limit	0.04198	0.01760	2.385
Random effects:	Variance	Std.Dev.	
Participant	0.9779	0.9889	

Table 11: Coefficients in a linear mixed-effects regression analysis predicting planning depth in the time pressure experiment treating time limit as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without time limit	311.47	318.97	-152.74	305.47			
With time limit	307.95	317.95	-149.97	299.95	5.5237	1	0.01876

Table 12: Likelihood ratio test between the model in table 11, and a restricted regression without the fixed effect of time limit.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.238714	0.019526	12.225
Time limit	0.002745	0.001018	2.698
Random effects:	Variance	Std.Dev.	
Participant	0.006002	0.07747	

Table 13: Coefficients in a linear mixed-effects regression analysis predicting feature drop rate in the time pressure experiment treating time limit as a continuous predictor and controlling for participant-specific effects using random intercepts.

we can write the number of non-terminal states as

$$N = \sum_{n=0}^{36} N_s \left(\left\lceil \frac{n}{2} \right\rceil, \left\lfloor \frac{n}{2} \right\rfloor \right) p_{\text{nt}} \left(\left\lceil \frac{n}{2} \right\rceil, \left\lfloor \frac{n}{2} \right\rfloor \right) \tag{6}$$

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without time limit	-185.86	-178.36	95.93	-191.86			
With time limit	-190.84	-180.84	99.42	-198.84	6.9792	1	0.008246

Table 14: Likelihood ratio test between the model in table 13, and a restricted regression without the fixed effect of time limit.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.0.5004489	0.0095997	52.132
Time limit	0.0008581	0.0005641	1.521
Random effects:	Variance	Std.Dev.	
Participant	0.001094	0.03308	

Table 15: Coefficients in a linear mixed-effects regression analysis predicting heuristic quality in the mobile data treating as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without time limit	-309.12	-301.62	157.56	-315.12			
With time limit	-309.43	-299.43	158.72	-317.43	2.3083	1	0.1287

Table 16: Likelihood ratio test between the model in table 15, and a restricted regression without the fixed effect of time limit.

Fixed effects:	Estimate	Std. Error	T value
Intercept	-69.75210	4.13493	-16.87
Games played	1.12853	0.04357	25.90
Random effects:	Variance	Std.Dev.	
Participant	8743	93.51	

Table 17: Coefficients in a linear mixed-effects regression analysis predicting Elo rating in the mobile experiment treating games played as a continuous predictor and controlling for participant-specific effects using random intercepts.

where $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote the ceiling and floor functions, respectively, $N_s(m, n)$ is the number of distinct configurations of m black and n white pieces on the board and $p_{\text{nt}}(m, n)$ is the

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
With games played	61400	61420	-30697	61394			
Without games played	60782	60808	-30387	60774	620.27	1	< 0.001

Table 18: Likelihood ratio test between the model in table 17, and a restricted regression without the fixed effect of games played.

Fixed effects:	Estimate	Std. Error	T value
Intercept	5.197720	0.072888	71.31
Games played	0.010814	0.001004	10.77
Random effects:	Variance	Std.Dev.	
Participant	0.8798	0.938	

Table 19: Coefficients in a linear mixed-effects regression analysis predicting planning depth in the mobile experiment treating games played as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without games played	22015	22034	-11004	22009			
With games played	21902	21928	-10947	21894	114.44	1	< 0.001

Table 20: Likelihood ratio test between the model in table 19, and a restricted regression without the fixed effect of games played.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.2950	0.003511	84.027
Games played	$-2.578 \cdot 10^{-4}$	$4.659 \cdot 10^{-5}$	-5.533
Random effects:	Variance	Std.Dev.	
Participant	0.002772	0.05265	

Table 21: Coefficients in a linear mixed-effects regression analysis predicting feature drop rate in the mobile experiment treating games played as a continuous predictor and controlling for participant-specific effects using random intercepts.

probability that a random configuration with that number of pieces does not contain 4-in-a-row patterns. The number of possible configurations is given by the multinomial coefficient

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without games played	-8553.7	-8534.1	4279.8	-8559.7			
With games played	-8582.2	-8556.1	4295.1	-8590.2	30.508	1	< 0.001

Table 22: Likelihood ratio test between the model in table 21, and a restricted regression without the fixed effect of games played.

Fixed effects:	Estimate	Std. Error	T value
Intercept	0.4774	0.003345	142.7
Games played	$6.115 \cdot 10^{-4}$	$4.245 \cdot 10^{-5}$	14.4
Random effects:	Variance	Std.Dev.	
Participant	0.003261	0.0571	

Table 23: Coefficients in a linear mixed-effects regression analysis predicting heuristic quality in the mobile experiment treating games played as a continuous predictor and controlling for participant-specific effects using random intercepts.

	AIC	BIC	Loglik	Deviance	χ^2	DoF	p
Without games played	-9084.6	-9065.0	4545.3	-9090.6			
With games played	-9284.9	-9258.9	4646.5	-9292.9	202.34	1	< 0.001

Table 24: Likelihood ratio test between the model in table 23, and a restricted regression without the fixed effect of games played.

$N_s(m,n) = \frac{36!}{m!n!(36-m-n)!}$, which we can evaluate analytically. To estimate the probability that a random state is non-terminal, we use a Monte Carlo procedure, generating random configurations and counting how many contain 4-in-a-row patterns. This results in an estimate of $1.1812 \cdot 10^{16} \pm 3.1 \cdot 10^{13}$ non-terminal states.

3 Model comparison

To validate the specification of our main model, we compare it to several alternatives. We test three categories of alternative models: *lesions*, generated by removing model components; *extensions*, generated by adding new model components; and *modifications*, generated by re-

placing a model component with a similar implementation.

In Fig S1A, we show the cross-validated log-likelihood per move of each model averaged across all participants in the human-vs-human, generalization, eye tracking, learning and time pressure experiments. All lesion models fit worse than the main model, showing that all model components are necessary to capture human behavior. In particular, lesioning any feature, tree search or feature dropping considerably worsens the model. Lesioning the active scaling constant also worsens the model but its effect is small. Of the modifications, the optimal weights and Monte Carlo Tree Search models perform poorly. The remaining models, as well as the model extensions, perform almost identically to the main model. In Fig S1B-F, we show the model comparison for each data set individually, with highly consistent results (correlation across models between experiments: $\rho = 0.932 \pm 0.010$).

In Figure 2, we show the performance of all models in different sessions of the learning experiment, and we find that the main model is among the best in all sessions. This demonstrates that experience-related changes in behavior are not due to experts using different models, but the same model with different parameters.

Together, these results show that people's choices are consistent with a broad class of planning algorithms, namely ones that contain a feature-based evaluation function, tree search, pruning and a mechanism to capture attentional oversights. We select our main model as a representative of this class that balances parsimony with predictive power. This model captures human behavior across a variety of experimental conditions: playing against other humans or computers, under time pressure, or while head-fixed with the experimenter present for eye tracking.

4 Reliability of parameter estimates and derived metrics

To assess the reliability of our parameter estimation methods, we perform two analyses. First, we compute the correlation between inferred parameters on two independent fits of the main

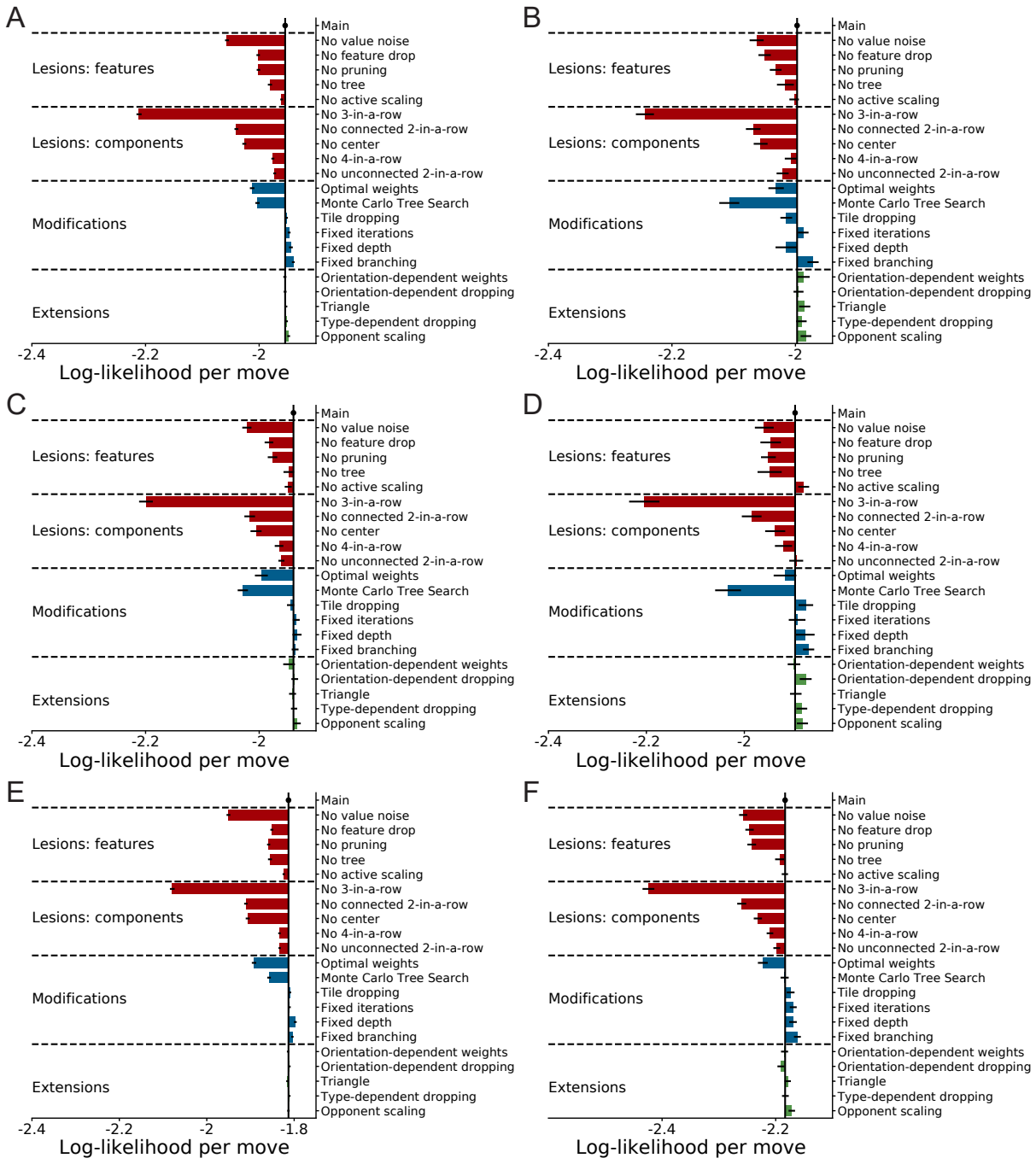

Figure 1: **Model comparison. A.** Cross-validated log-likelihood per move, averaged across all participants in the laboratory experiments, for the main model and its alternatives. Error bars indicate standard error of the mean log-likelihood difference between a model and the main model. **B.** Same as **A.**, for participants in the human-vs-human experiment only. **C-F.** Same, for the generalization, eye tracking, learning and time pressure experiments.

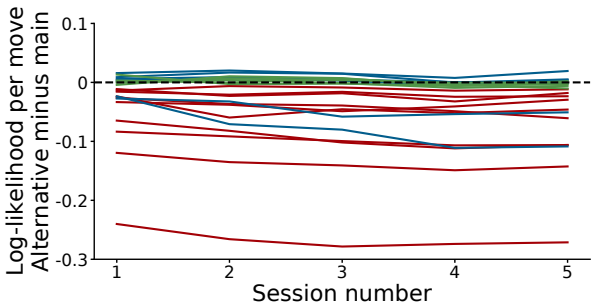

Figure 2: Difference between log-likelihood per move of alternative models minus that of the main model, as a function of session number in the learning experiment. The color indicates the model type (lesion, modification or extension). We observe no indication that any alternative models consistently outperform the main model on early or later sessions in the experiment.

model to the human-vs-human data across participants (Fig S3A). In principle, since we use maximum-likelihood estimation, the best-fitting parameters are a deterministic function of the data. However, because IBS only provides stochastic estimates of the log-likelihood, we also obtain stochastic estimates of the maximum-likelihood parameters. We find that all parameters except the lapse rate are correlated across the two independent fits, and the most reliably estimated parameters are w_{center} ($\rho = 0.93, p < 0.001$), $w_{\text{connected 2-in-a-row}}$ ($\rho = 0.92, p < 0.001$) and δ ($\rho = 0.74, p < 0.001$).

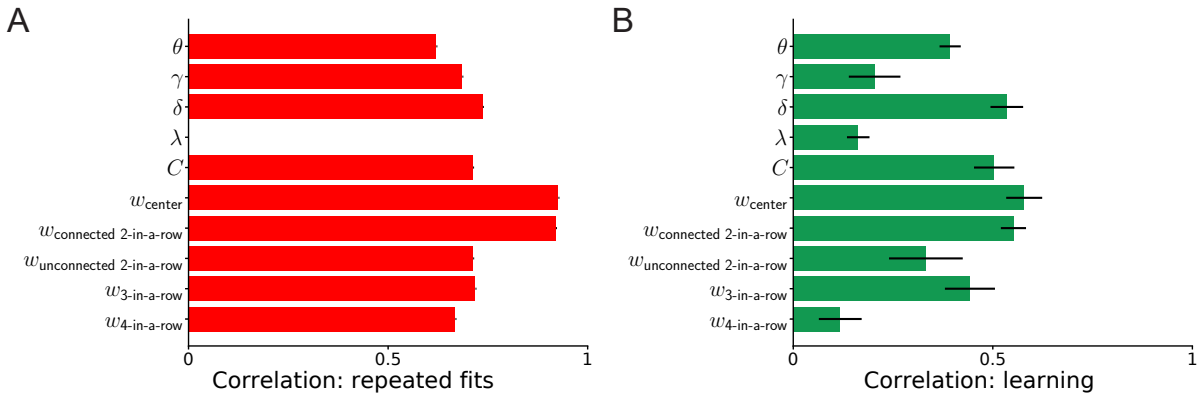

Figure 3: Pearson correlation across participants between model parameters estimated in **A.** two independent fits, **B.** different sessions in the learning experiment. The errorbars denote standard error of the mean correlations across all session pairs.

Second, we compute the correlation between parameter estimates for the same participant on different sessions of the learning experiment (Fig S3B). This correlation assesses both noise induced by the stochastic nature of the model fitting methodology, but also the reliability of people's strategy across time. We average this metric across all $(5 \times 4)/2 = 10$ combinations of the 5 learning sessions. The correlations are positive for all parameters, with once again the most reliable parameters being w_{center} ($\rho = 0.578 \pm 0.045, t(9) = 12, p < 0.001$), $w_{\text{connected 2-in-a-row}}$ ($\rho = 0.551 \pm 0.031, t(9) = 17, p < 0.001$) and δ ($\rho = 0.535 \pm 0.041, t(9) = 12, p < 0.001$).

These analyses show that the model parameters are estimated reliably, but our main results depend on the derived metrics of planning depth, feature drop rate and heuristic quality instead of the raw model parameters. Therefore, we repeat these reliability analyses and show that planning depth and heuristic quality are both reliably estimated on independent fits (planning depth: $\rho = 0.86, p < 0.001$, heuristic quality: $\rho = 0.77, p < 0.001$, Fig S4A), and different sessions of the learning experiment (planning depth: $\rho = 0.508 \pm 0.044, t(9) = 12, p < 0.001$, heuristic quality: $\rho = 0.544 \pm 0.051, t(9) = 10, p < 0.001$, Fig S4B). Note that since the feature drop rate is simply the parameter δ, this is already tested in Fig S3.

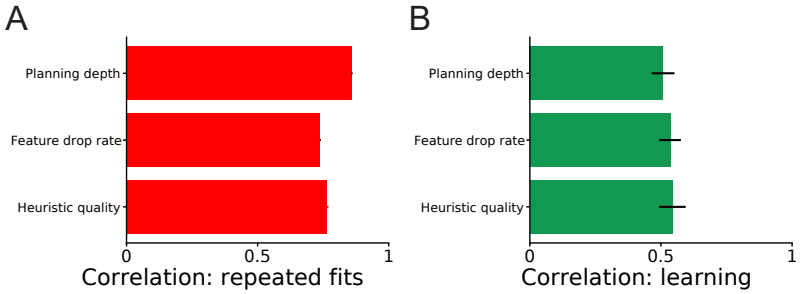

Figure 4: Same as Fig S3, for the derived metrics.

5 Trade-offs between estimated model parameters

Aside from noise and uncertainty in parameter estimates, one may worry about parameter identifiability and trade-offs. A trade-off occurs when two (or more) parameters in a model cause similar changes in the distribution of responses, and therefore inferring which parameter is responsible for a given shift in behavior is difficult. One method to assess trade-offs is a parameter recovery analysis, in which one fits the model on fake data sets generated by varying one parameter, and asks if this causes shifts in the estimated values of other parameters. However, as mentioned in S2.6, the model fitting pipeline is slow, and a parameter recovery analysis is intractable. Instead, we investigate trade-offs by analyzing parameter estimates in the lesion models. In each lesion model, a specific parameter θ_i is fixed to a constant. If θ_i trades off with θ_j, the optimization algorithm will compensate for the lesioning of θ_i by increasing (or decreasing) θ_j. Specifically, we test for two signatures of trade-offs:

1. Fixing θ_i to a constant changes the distribution (across participants) of $\hat{\theta_j}$.

2. The change in $\hat{\theta}_j$ that results from fixing θ_i is correlated (across participants) with the value of $\hat{\theta}_i$ in the unconstrained model.

The first signature is generic and tests for any kind of non-independence of θ_i and θ_j. The second signature specifically tests for those kind of trade-offs that could complicate the interpretation of correlations between $\hat{\theta}_j$ and experimental variables such as session number or time limit condition. Namely, if such a trade-off is present, and there exists a true correlation between that experimental variable and θ_i, but the model fitting pipeline for some reason produces biased estimates of θ_i, it might compensate for this bias by increasing θ_j. This could then lead to a spurious correlation between the experimental variable and θ_j. To detect these signatures, we perform two tests for for each (θ_i, θ_j) pair:

1. a 2-sample Kolmogorov-Smirnov (KS) test between the distribution of $\hat{\theta}_j^{\text{lesion } i}$ (the j-th parameter in the model where parameter i is lesioned) and $\hat{\theta}_j^{\text{full}}$ across participants in the human-vs-human experiment (see Fig S5A)

2. the Pearson correlation between $\hat{\theta}_i^{\text{full}}$ and $\hat{\theta}_j^{\text{full}} - \hat{\theta}_j^{\text{lesion } i}$ (see Fig S5B).

In both of these tests, we compensate for multiple comparisons using a false discovery rate of $\alpha_{\text{FDR}} = 0.05$.

The KS test reveals a mutual interaction between the 4-in-a-row feature weight and pruning, and unidirectional effects of lesioning the three-in-a-row feature or the decision tree on almost every other parameter. Additionally, the active scaling constant is affected by lesioning the feature drop mechanism or the connected two-in-a-row feature. The Pearson correlation test only reveals interactions between the feature drop rate, active scaling constant, connected two-in-a-row and 3-in-a-row feature weights. However, the only trade-off for which both tests are significant is the effect of lesioning the 3-in-a-row feature on the feature drop rate.

These results show that parameters in our model are not fully distinguishable, which is a consequence of the model fitting methodology as well as the finite data size. However, the main results of our paper concerns the derived metrics of planning depth, feature drop rate and heuristic quality. To show that these metrics are distinguishable, we analyze the **No tree**, **No feature drop** and **Optimal weights** lesion models. In these models, the planning depth, feature drop rate or heuristic quality are constant, hence we can think of these as 'lesions' of those metrics. As before, we test whether the distribution of planning depth, feature drop rate or heuristic quality in the lesion models differs from that in the main model, and whether the change induced by the lesion correlates with the value of the lesioned metric. Fig S6A shows the result of the Kolmogorov-Smirnov test, Fig S6B the Pearson correlation. We find no significant trade-offs that could lead to spurious effects in our analyses of planning depth, feature drop rate

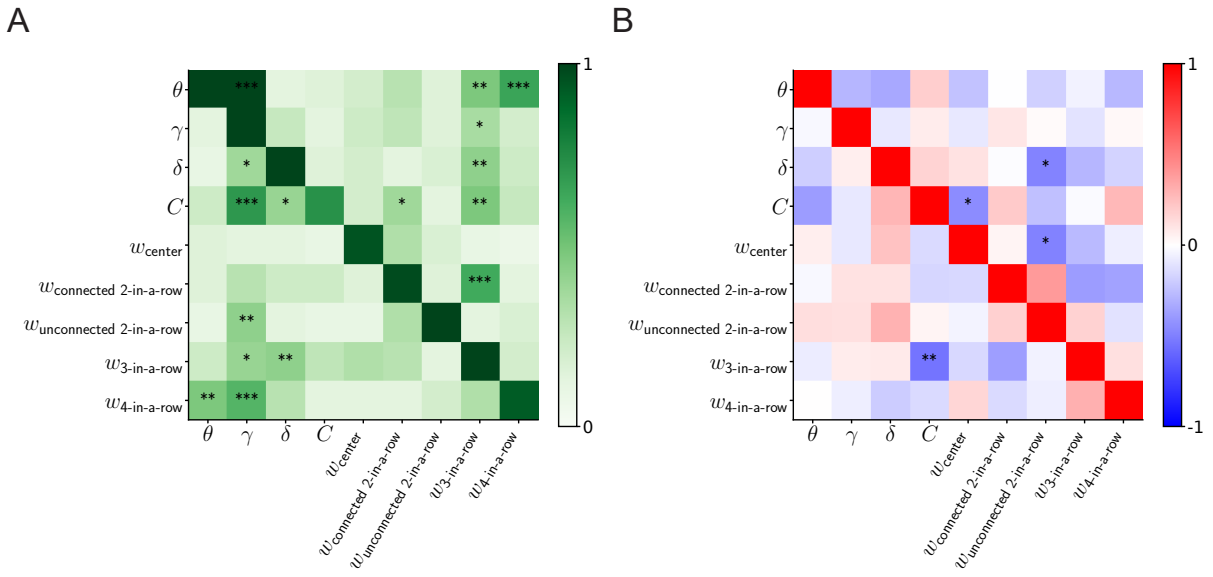

Figure 5: We detect trade-offs between model parameters using two approximate methods, since computing the full confusion matrix in a parameter recovery analysis is too computationally costly. **A.** 2-sample Kolmogorov-Smirnov (KS) test statistic between the distribution of $\hat{\theta}_j^{\text{lesion } i}$ and $\hat{\theta}_j^{\text{full}}$ for each pair of parameters. We indicate statistics that are significant at $\alpha = 0.05$, $\alpha = 0.01$, $\alpha = 0.001$ (all FDR-corrected) with one, two or three stars, respectively. **B.** Pearson correlation between $\hat{\theta}_i^{\text{full}}$ and $\hat{\theta}_j^{\text{full}} - \hat{\theta}_j^{\text{lesion } i}$ for each pair of model parameters, with significance levels as in **A**.

and heuristic quality as a function of expertise or time pressure.

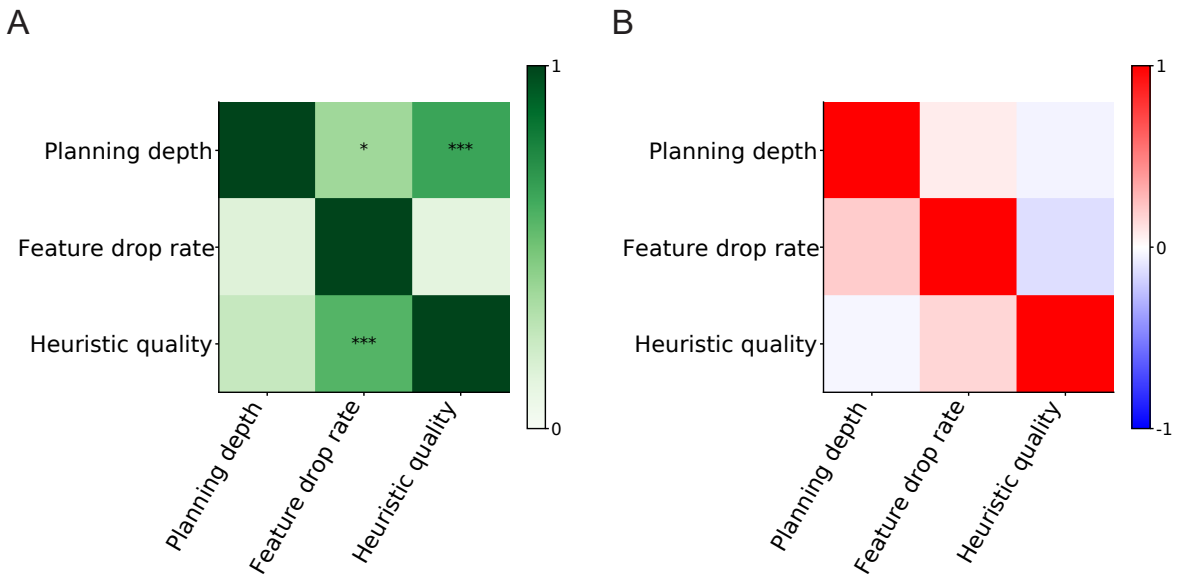

Figure 6: Same as Fig S5, but for derived metrics.

6 Summary statistics

We would like to directly compare the main model to human choices, which is challenging because the data (moves in board positions) is high-dimensional and discrete. Instead, we compare summary statistics. For each move made by each human player, we compute the distance from the chosen square to the center of the board, the distance to the nearest piece owned by that player, the distance to the nearest piece of the opponent, the distance to the the center of mass of that player's pieces, the distance to the center of mass of the opponent's pieces, the number of that player's pieces on the 8 squares neighboring the chosen square, and the number of opposing pieces on neighboring squares. Additionally, we indicated whether with their chosen move, the player created a threat to make 4-in-a-row on the next move or parried a direct threat from their opponent. Together, this yields 9 summary statistics. We also

compute these statistics for moves made by the main model in the same positions encountered by human players, with parameters inferred for those players, and for random moves.

Fig S7 shows the average of these 9 summary statistics, aggregated across all participants in the human-vs-human data set, as a function of the number of pieces on the board. This analysis probes systematic patterns in the time course of people's games, for example a tendency to start playing near the center of the board and gradually expand outwards. For all summary statistics, people deviate considerably from random, and the main model matches the human data almost exactly. In Fig S8, we plot the average of these 9 summary statistics for each participant, against the prediction of the main model. The human data and model prediction are highly correlated for all summary statistics.

7 Example positions illustrating model components

To investigate which patterns in the data are explained by different model components, we compare the distribution of choices predicted by the main model and different lesion models. First, we compare the main model to the **No tree** model. We simulate moves from both models in all positions that occurred in human-vs-human games, for 200 different parameter vectors taken from previous model fits. We then average the move distributions across those 200 parameter vectors, and calculate the Jensen-Shannon divergence between the main model and that of the no tree lesion. This JS divergence is high for positions in which the model makes highly different moves with and without planning, irrespective of the parameters. In other words, such positions maximally distinguish planning vs no planning.

Fig S9 shows 5 example positions for which this dissimilarity is among the highest (the maximum is 0.149). Upon inspection, we recognize these positions as ones where the player to move has multiple reasonable options, but to evaluate their quality one has to calculate many moves ahead. For example, in the second position, the move preferred by the **No tree** model

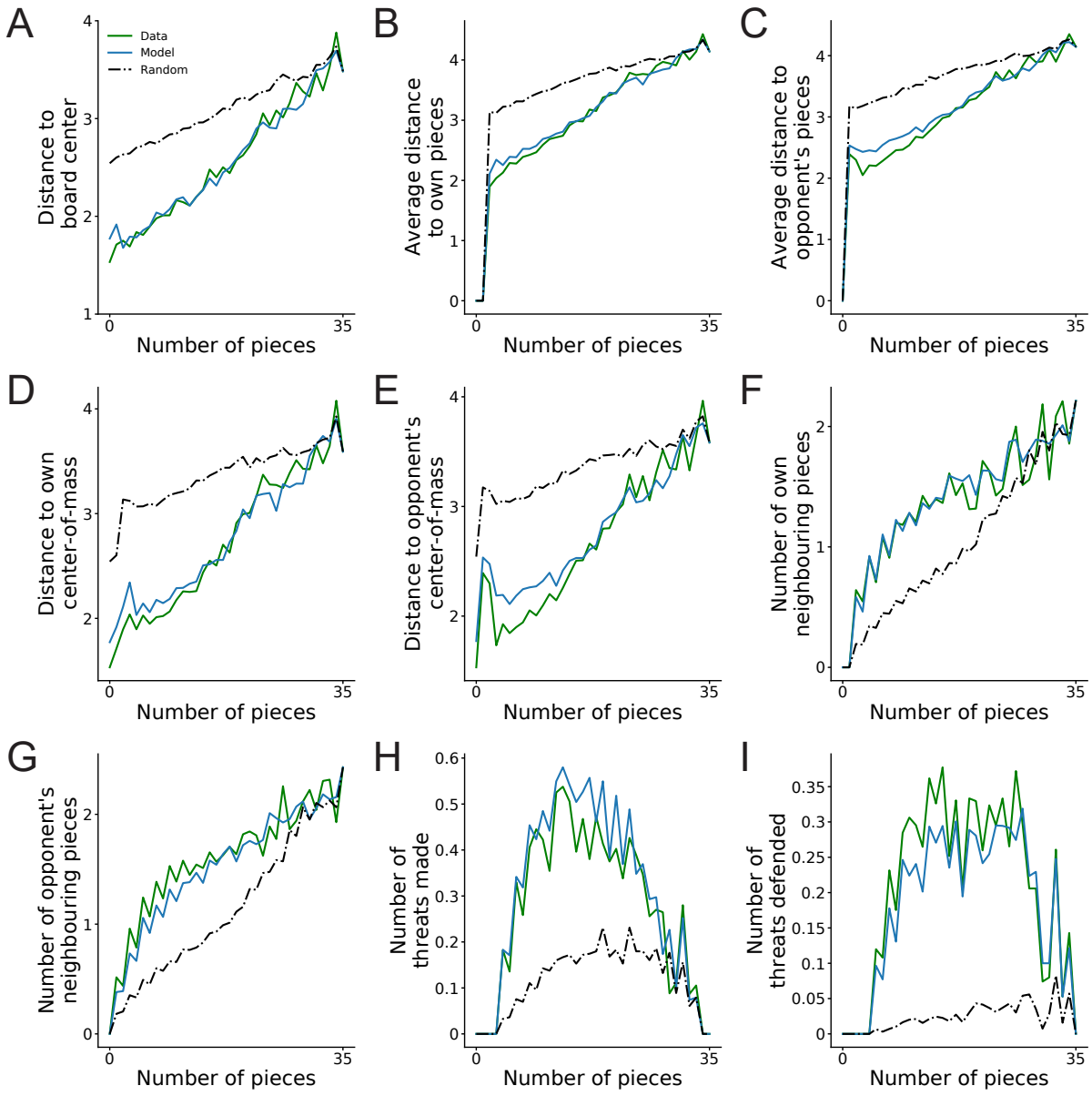

Figure 7: Summary statistics computed as a function of number of pieces on the board, for moves made by human players (green solid lines), the behavioral model with inferred parameters (blue lines) or random moves (black dashed lines). These graphs depict cross-validated predictions. For all statistics, the model prediction is much closer to the human data than random.

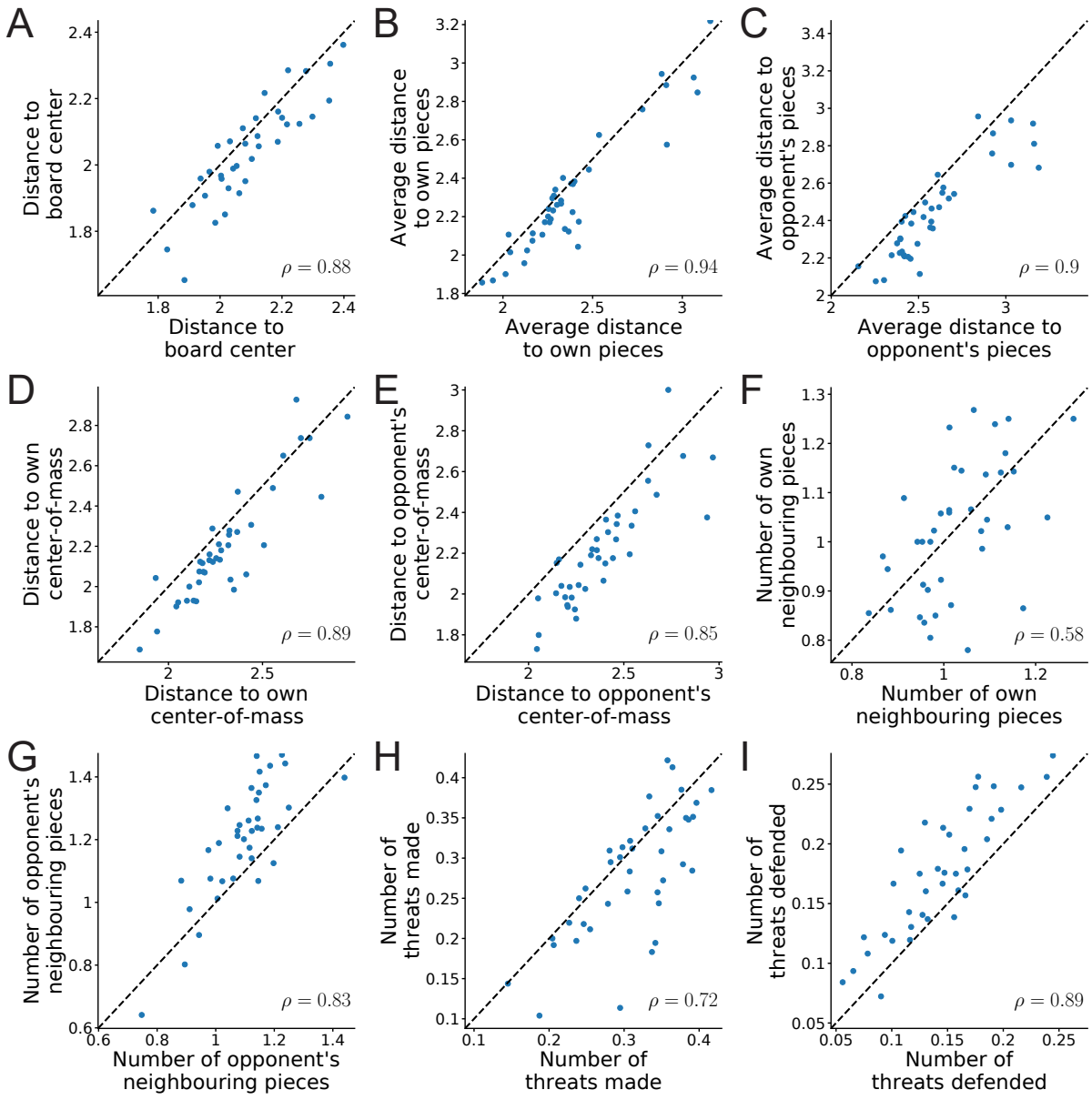

Figure 8: The model predicts individual differences in summary statistics. Each panel shows a scatterplot for a summary statistic, where each point represents a participant in the human-vs-human experiment, the x-coordinate the statistic computed on that participant's moves and the y-coordinate the statistic computed on moves made by the model. The inset in each panel shows the corresponding Pearson correlation coefficient.

is losing and the one by the main model is drawn, but this relies on a specific 10-move forced sequence that can only be found through explicit search.

Next, we repeat the analysis but compare the main model against the **Optimal weights** lesion model. This highlights positions in which human preferences do not align with the optimal value function, and motivate the need to fit feature weights to human behavior rather than using a fixed value function. Fig S10 shows 5 example positions. One common pattern in these positions is that the optimal weight model often prefers making a direct 3-in-a-row threat over moves that set up multiple 2-in-a-row threats. Additionally, the optimal weights is often overly confident in its predictions, which suggests that it is unable to capture human errors that stem from incorrect feature weights.

To investigate the contribution of different sources of noise (feature dropping and value noise), we perform a version of the same analysis, except as our metric we use the ratio between the probability of the human move in the main model and the model without that noise source. In other words, this analysis highlights moves made by a human player that can only be explained using the different noise sources.

Fig S11 shows 5 example positions for feature dropping. The feature drop mechanism is necessary to explain a human tendency to overlook possibilities to immediate make 4-in-a-row, or immediate 4-in-a-row threats by the opponent. More generally, the feature drop mechanism applies to all features in the model, but the impact is largest when considering 3-in-a-row and 4-in-a-row features.

Fig S12 shows 5 examples in which the human move cannot be explained without value noise. These positions are less clearly related, but one common pattern is that the human move has close to zero probability according to the model without value noise, and the value noise mechanism essentially acts as a mechanism to smooth out the distribution.

Finally, we investigate which moves necessitate a non-zero lapse rate in the model. For

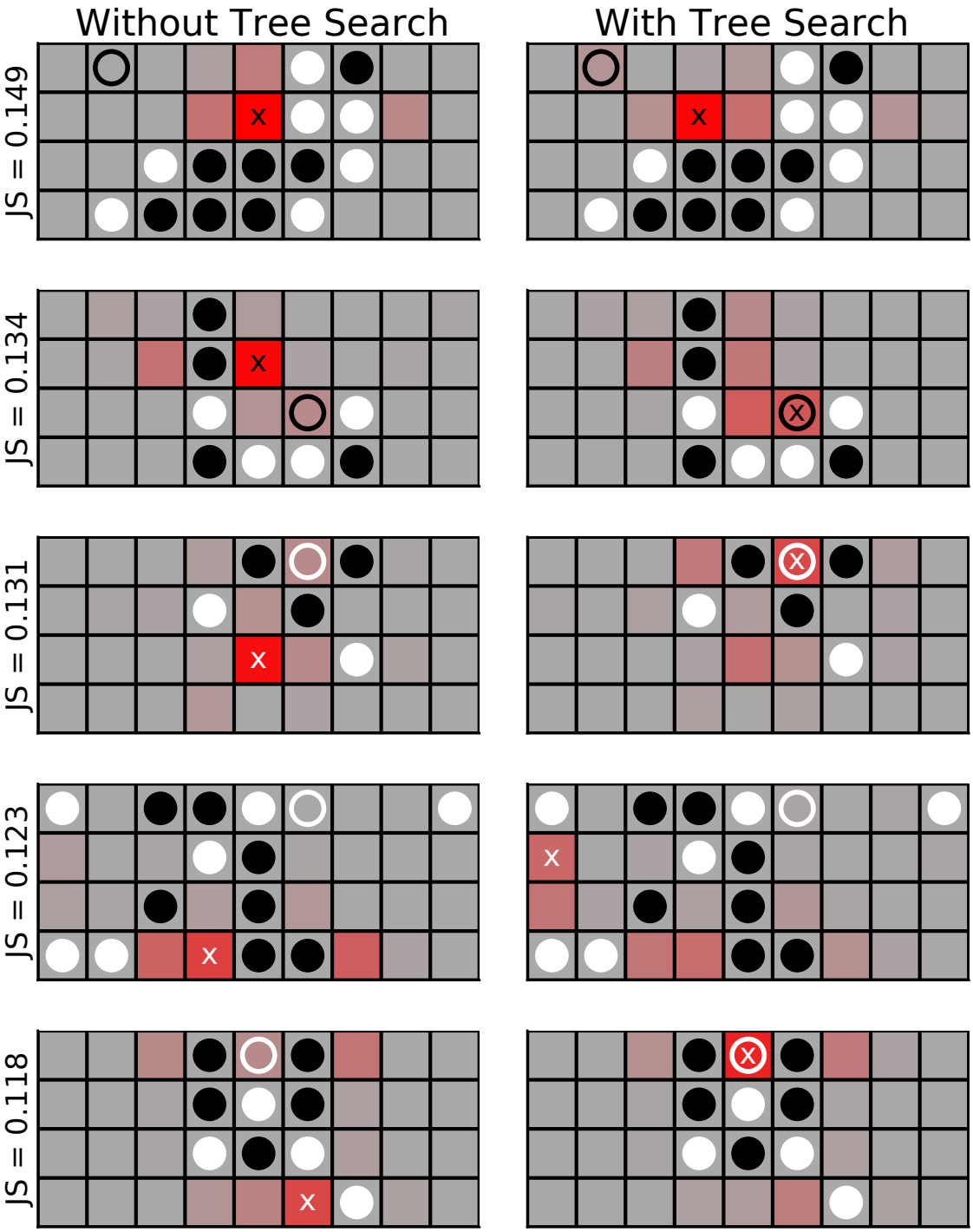

Figure 9: Example positions from human-vs-human games in which the model with (right column) and without tree search (left column) make highly different predictions (red shade). In each position, we also show the models' preferred move (with an x) and the move made by the human participant (open circle).

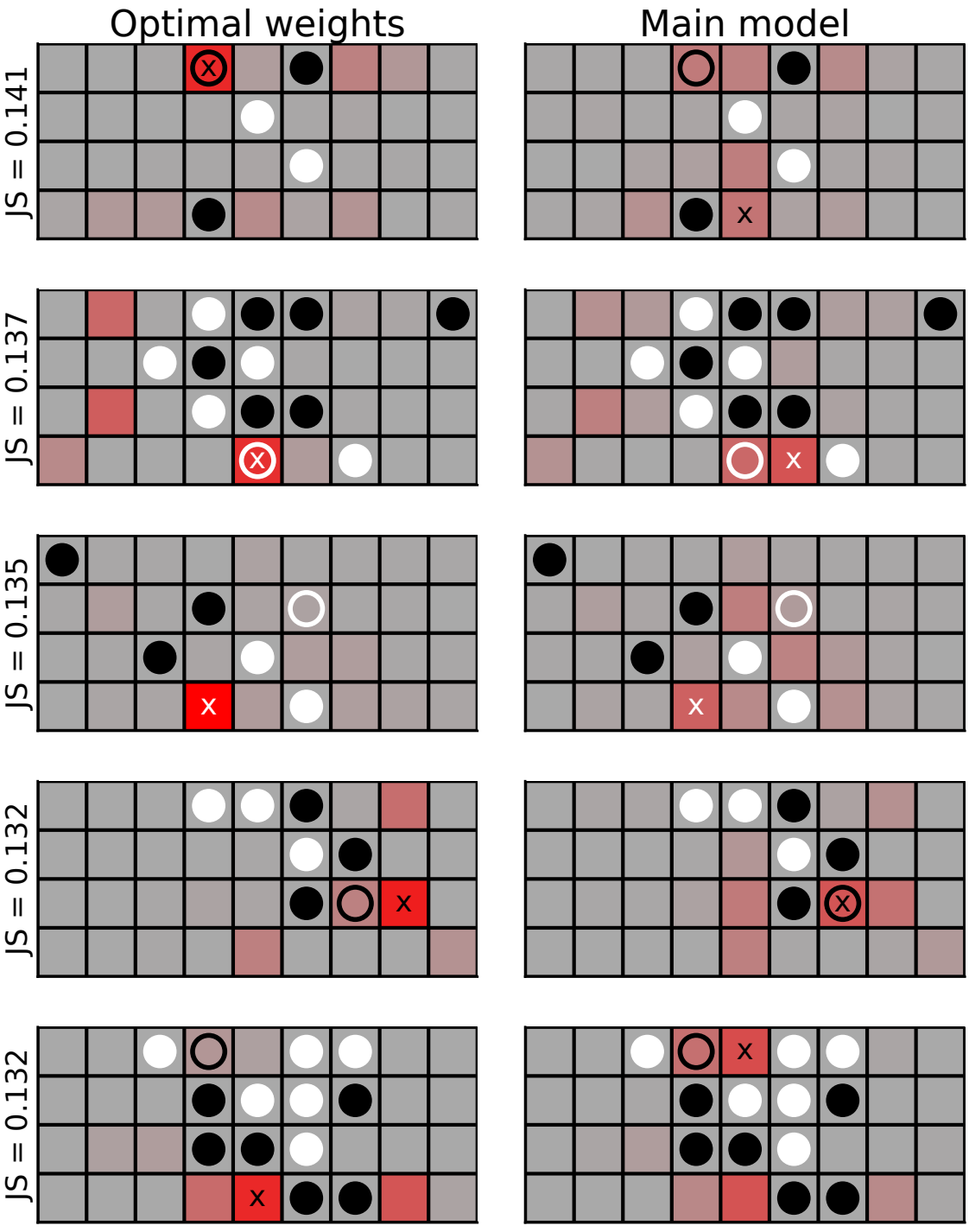

Figure 10: Same as Fig S9, for the **Optimal weights** model compared to the main model.

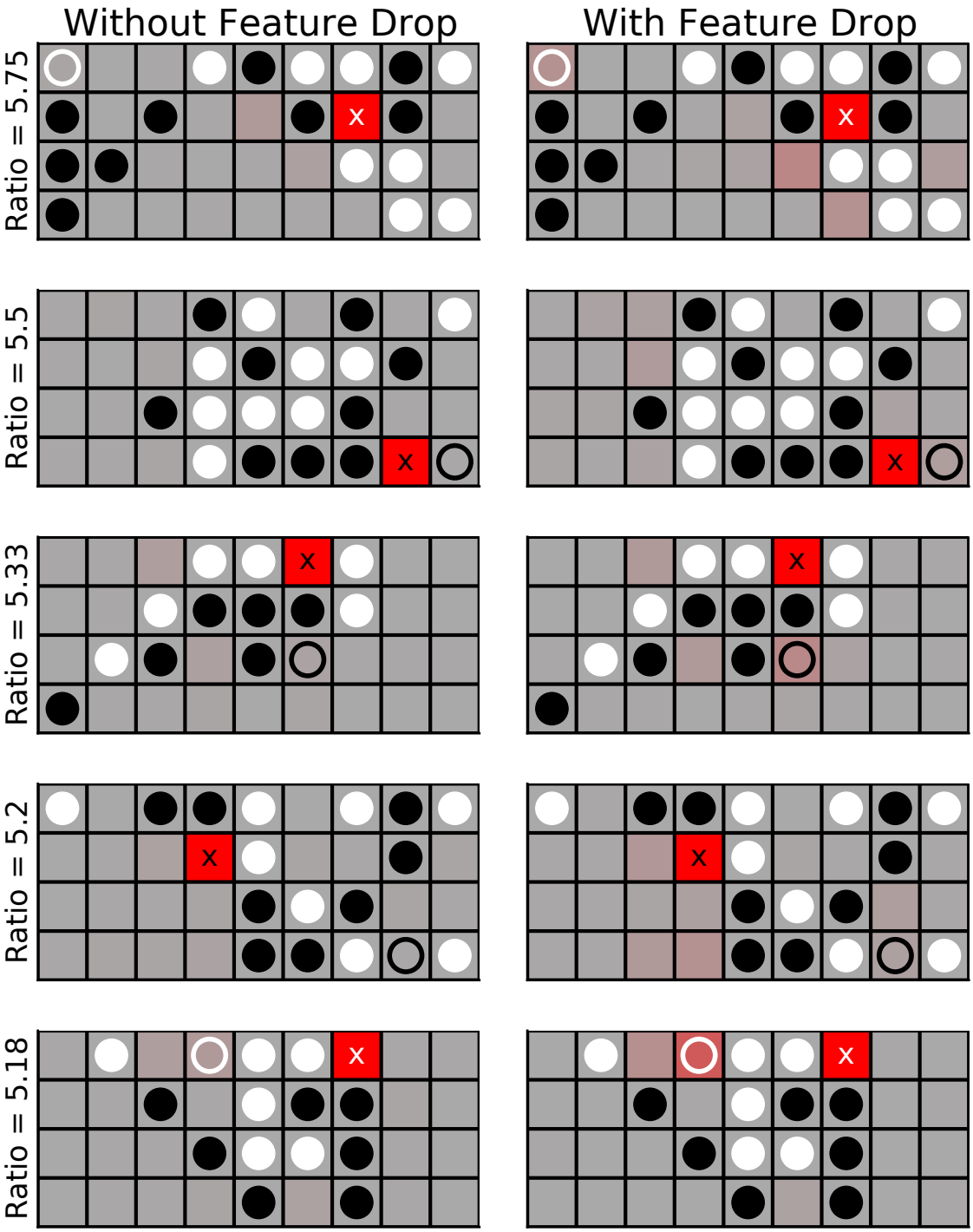

Figure 11: Same as Fig S9, but comparing the main model with the **No feature dropping** lesion, and using the ratio between the predicted probability of the human move (circle) as the metric (red shading).

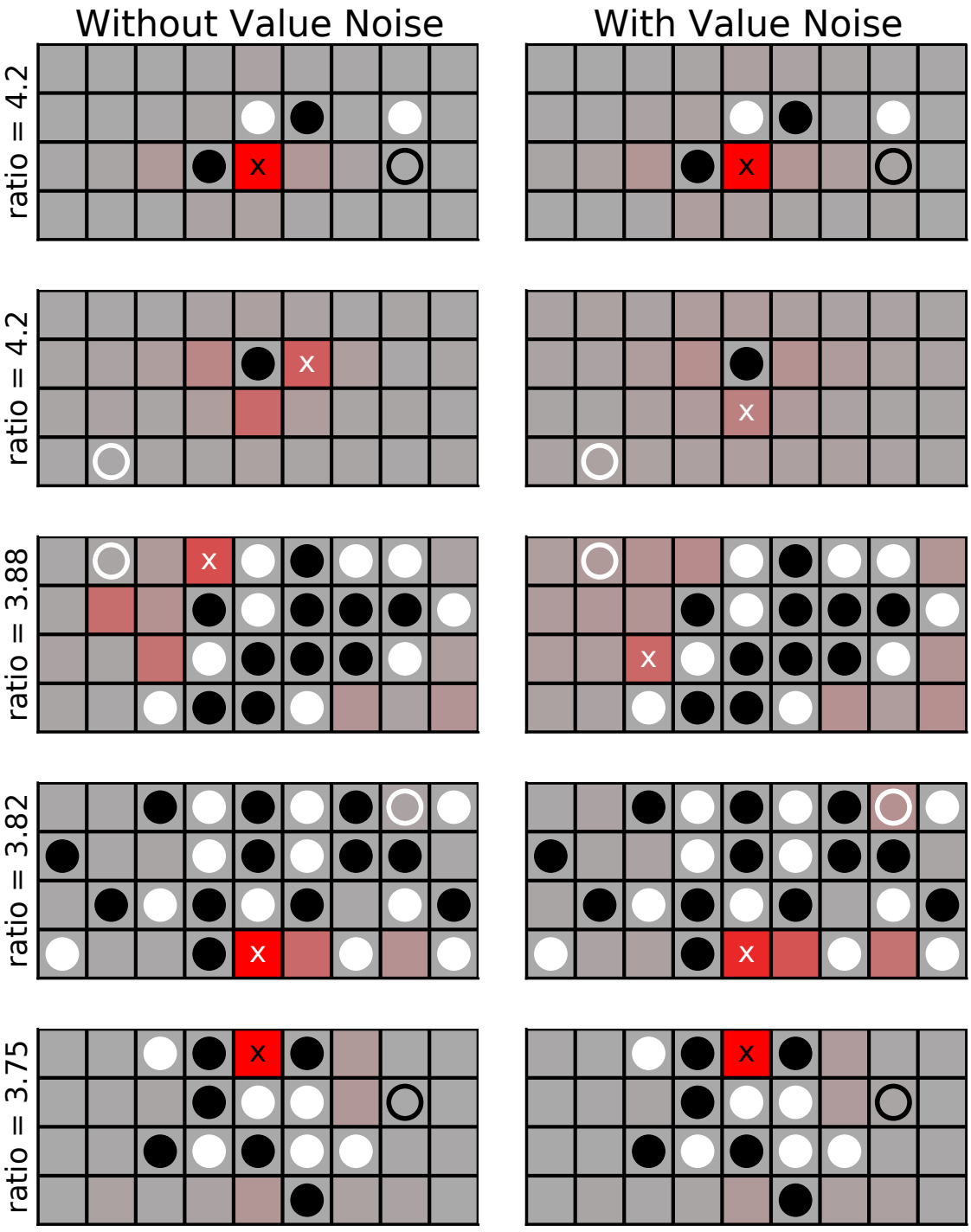

Figure 12: Same as Fig S9, but comparing the main model with the **No value noise** lesion.

this analysis, we cannot fit a **No lapse** model, but that is numerically unstable. Instead, we run 200 simulations of the main model with different parameter vectors (see above), and as our qualifying metric we use the probability of the human move averaged across these 200 simulations. Fig S13 shows the top-5 positions where the human move is least probable. These positions are all instances where the human participant made a particularly poor (and in 4 out of 5 cases, losing) move that has no apparent rationale behind it. Thus, the model can only explain these choices as lapses, i.e., the human participant making a completely random move.

8 2AFC comparison with an oracle model

Here, we show that the main model predicts people's 2AFC and evaluation decisions better than an oracle model, which has access to the game-theoretic value of each position and the objectively correct moves (S2.4). On 2AFC trials where one move is objectively better than the other, the oracle model picks that move. On trials where both moves are equally strong, the model picks randomly. On evaluation trials, the oracle model responds 1 when the position is lost, 4 when it's a draw and 7 for winning positions.

In Fig S14, we show the performance of this oracle model against that of the main model. For both 2AFC and evaluation, the main model outperforms the oracle (2AFC: $\Delta\text{acc} = 0.0362 \pm 0.0093, t(39) = 3.8, p < 0.001$, evaluation: $\Delta\rho = 0.079 \pm 0.012, t(39) = 6.5, p < 0.001$). These results suggests the main model predicts subjective preferences of individual participants.

9 Turing test

In the Turing test, we showed participants video segments of sequences of moves. Figure 15A shows observers' classification accuracy as a function of the length of the video, that is, the number of moves in the segment. On average, sequences were 9.38 moves long. Showing sequences instead of individual moves was necessary, as participants are not above chance for classifica-

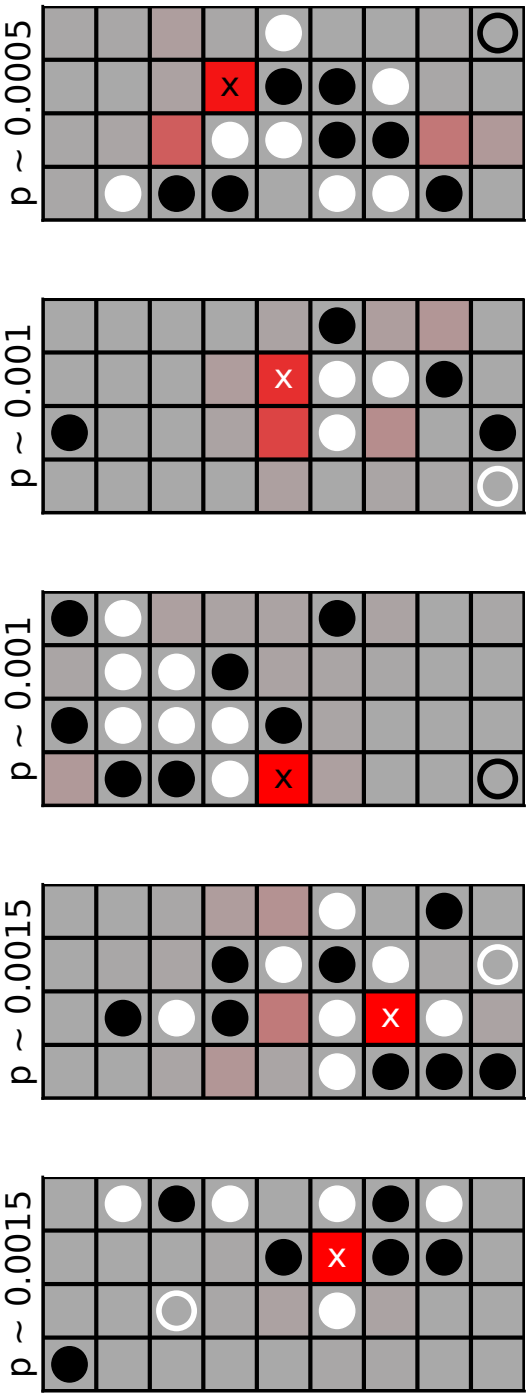

Figure 13: Same as Fig S9, but only showing the main model, and using the probability of the human move (circle) as the metric (red shading).

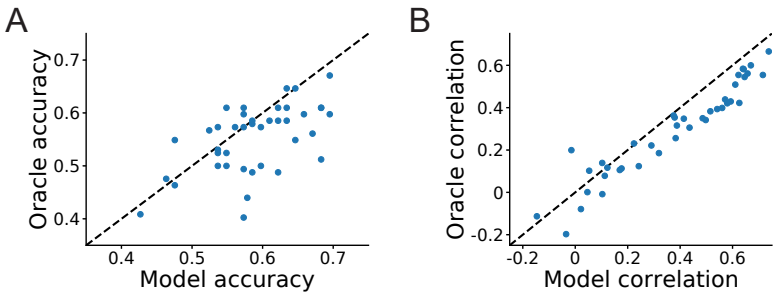

Figure 14: **A.** Accuracy of the main model prediction and that of an oracle model on 2AFC data. Each point is a participant, the dashed line indicates equality. **B.** Same, for the Pearson correlation coefficient between predicted and observed board evaluations. For both 2AFC and evaluation, the main model outperforms the oracle.

tion of one-move videos (of which there were 8). Instead, participants require longer sequences and their accuracy only substantially exceeds 50% for sequences longer than 10 moves. A mixed effects linear regression with accuracy as dependent variable and observer-specific random intercepts estimates the increase in accuracy per observed move as only $0.33 \pm 0.10\%$.

For each video, we calculate the percentage of observers who classify it as being generated from a human-vs-human game. Figure 15B shows the histogram of this percentage across videos, split up by the true label of the video (humans or computers). While it is true that human games are on average more likely to be classified as human and vice versa, there are no videos for which all 30 observers agree, and there is a considerable fraction of videos (63 out of 180) for which a majority of observers respond incorrectly.

10 Stopping rule

To make predictions for response times and eye movements, we amend the model with a stopping rule: we terminate the best-first search algorithm when the model's preferred move in the root node remains unchanged for 50 consecutive iterations. This stopping rule is in addition to the random stopping rule implied by the stopping probability γ. To obtain a response time

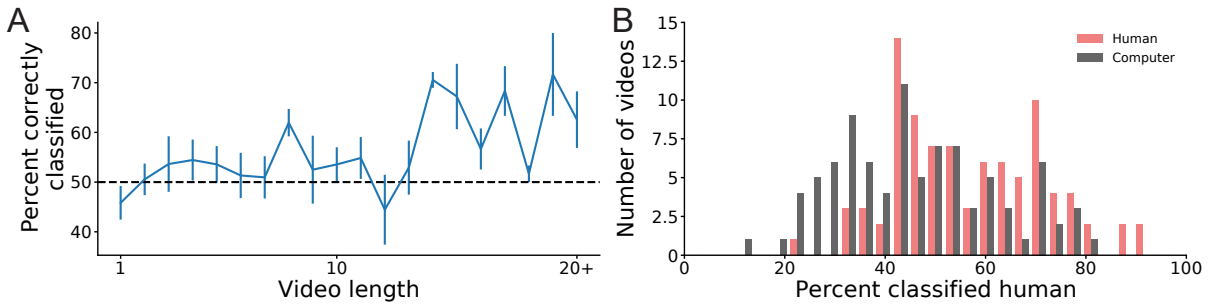

Figure 15: **A.** Classification accuracy in the Turing test as a function of video length. Error bars denote mean and standard error. **B.** Histogram of the percentage of observers classifying a given video as human-vs-human or computer-vs-computer, for either human games (pink), or computer-generated games (gray).

prediction from the model with given parameters on a given trial, we simulate 100 moves from the model, and measure the average number of search iterations executed before the algorithm terminates.

The stopping rule allows the model to predict differences in response times across trials for the same participant. In positions where one move is clearly preferred over other options (for example, if the opponent has a direct threat that needs to be parried), the algorithm will quickly rule any alternatives and spend its remaining time calculating the future consequences of that move. Therefore, the preferred move in the root node remains unchanged for many iterations, and the stopping rule causes the search to terminate, resulting in a low number of iterations. In other positions with many plausible alternatives, the model will waver, and the stopping rule will not be called and the model will search longer. Thus, the termination rule is one way of incorporating the known effect of decision difficulty on response times. Other plausible stopping rules might behave similarly. For example, one can base the termination criterion on relative node values or node visits, as in MCTS.

In the main text, we showed that the model predicts people's response times on individual trials with this stopping rule based on 50 iterations. However, the choice to use 50 iterations as

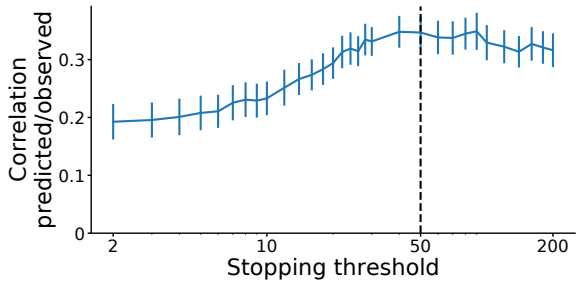

Figure 16: Correlation between predicted and observed log response times in the main model with different choices for the stopping threshold. The vertical dashed line at 50 indicates the stopping threshold used in the main text. The correlation is robust to the choice of stopping threshold, within a range from approximately 30 to 100.

the threshold is arbitrary. We therefore analyze models with stopping thresholds varying from 2 to 200 iterations (Fig S16). The correlation between predicted and observed response times is high for a broad range of stopping thresholds (30-100). Note that the lower the threshold, the larger the deviation between the choices made by the main model with and without the stopping rule. This analysis demonstrates that the correlation between search iterations in the model and participants' response times on individual trials is robust to the value of the stopping threshold, within a reasonable range.

11 Eye movement analyses

To pre-process eye tracking data, we first convert the output files from `edf` to `ASCII` format using the `edf2asc.exe` program provided by SR Research. This results in a time series of fixations and saccades for each participant, with time measured relative to the moment we turned on the eye tracking software, and position in a coordinate system defined by the eye tracker calibration. We start by re-calibrating both time and space, using data from the calibration that each participant completes at the beginning and end of the experiment (S1.5). In each calibration, the participant fixates on all 36 squares consecutively, pressing the space bar to

provide a time stamp for each fixation.

In Fig S17A, we plot the horizontal and vertical eye position as a function of time, for both calibration periods and one example participant. Note the typical staircase-like pattern that arises when a participant scans each row of the board, starting at the bottom, from left to right. Fig S17A also shows the time of each of the 72 space bar presses as vertical lines. To correct for small temporal offsets between times measured on the eye tracking computer and the experiment server, we varied a temporal offset between the behavioral and eye tracking time stamps and maximized the Pearson correlation between the eye coordinate at the (corrected) times of the space bar press and the corresponding tile coordinate, summed across horizontal and vertical dimensions. We then calibrated spatial coordinates using the temporally aligned data.

To recalibrate space, we assume the transformation from eye tracker to board coordinates is linear for each square, but not necessarily globally linear. We divide the eye coordinate space into 36 tiles using a Voronoi tessellation with as centroids the eye tracker coordinates at the time of space bar presses, averaged across the pre-task and post-task calibration (Fig S17B). The Voronoi tessellation allows us to assign a square for eye fixation in the time series.

For any fixation vector $\vec{c}$ in eye tracking coordinates, we compute the fractional board coordinates with a 3-step process.

1. Use the Voronoi tessellation to assign the fixation to a board square, and define $\vec{c}_{\text{tile}}$ to be the eye tracker coordinates of that tile's centroid.

2. For each neighboring square (up, down, left or right) with centroid coordinates $\vec{c}_{\text{neighbor}}$, compute

$$f_{\text{neighbor}} = 1 + \frac{\|\vec{c} - \vec{c}_{\text{tile}}\|^2}{\|\vec{c}_{\text{tile}} - \vec{c}_{\text{neighbor}}\|^2} - \frac{\|\vec{c} - \vec{c}_{\text{neighbor}}\|^2}{\|\vec{c}_{\text{tile}} - \vec{c}_{\text{neighbor}}\|^2} \tag{7}$$

This equation defines a linear function from the Voronoi cell to $[-\infty, 1]$, so that the centroid maps to 0 and any point on the boundary with this neighbor maps to 1.

3. Define the fractional coordinates in board space as

$$x = \begin{cases} f_{\text{right}}/2 & \text{if } |f_{\text{right}}| < |f_{\text{left}}| \\ -f_{\text{left}}/2 & \text{otherwise} \end{cases} \qquad y = \begin{cases} f_{\text{up}}/2 & \text{if } |f_{\text{up}}| < |f_{\text{down}}| \\ -f_{\text{down}}/2 & \text{otherwise} \end{cases} \tag{8}$$

For squares on the edge of the board, where f_{right}, f_{left}, f_{up} or f_{down} may be ill-defined, we use the case in the equation above that is well-defined. This results in a piecewise linear mapping between board space and eye tracker coordinates. In Fig S17C,D, we show the resulting mapping from eye tracker space to board coordinates for an example participant.

To predict eye movements from the behavioral model, we simulate 100 moves from the model in each position encountered by each participant in the eye tracking data set, with model parameters inferred for that participant. After the model makes a move, we count for each square and each depth how often the principal variation in the selection step of the search algorithm includes a move on that square at that depth. This results in a histogram over unoccupied squares for each position in our data set and each depth, which we normalize into probability distributions of the model's square visits. For each position, we also create a probability distribution that is 1 for the square on which the participant actually moved and zero otherwise. We then regress the model's square visit distributions at different depths and the actual move distribution against participants' distribution of attention estimated from eye movements. This results in a positive correlation between predicted and observed eye movements, as outlined in the main text. Crucially, the regression coefficients are significantly larger than zero (one-sample T-test across participants) for depth up to 7, and highest for depth closer to 1 (Fig S18).

To support the interpretation of dropped features as lapses of selective attention, we now show that when the behavior modeling suggests that a participant has dropped a feature, that participant spends little to no time looking at the relevant squares. Fig S19 shows 5 example

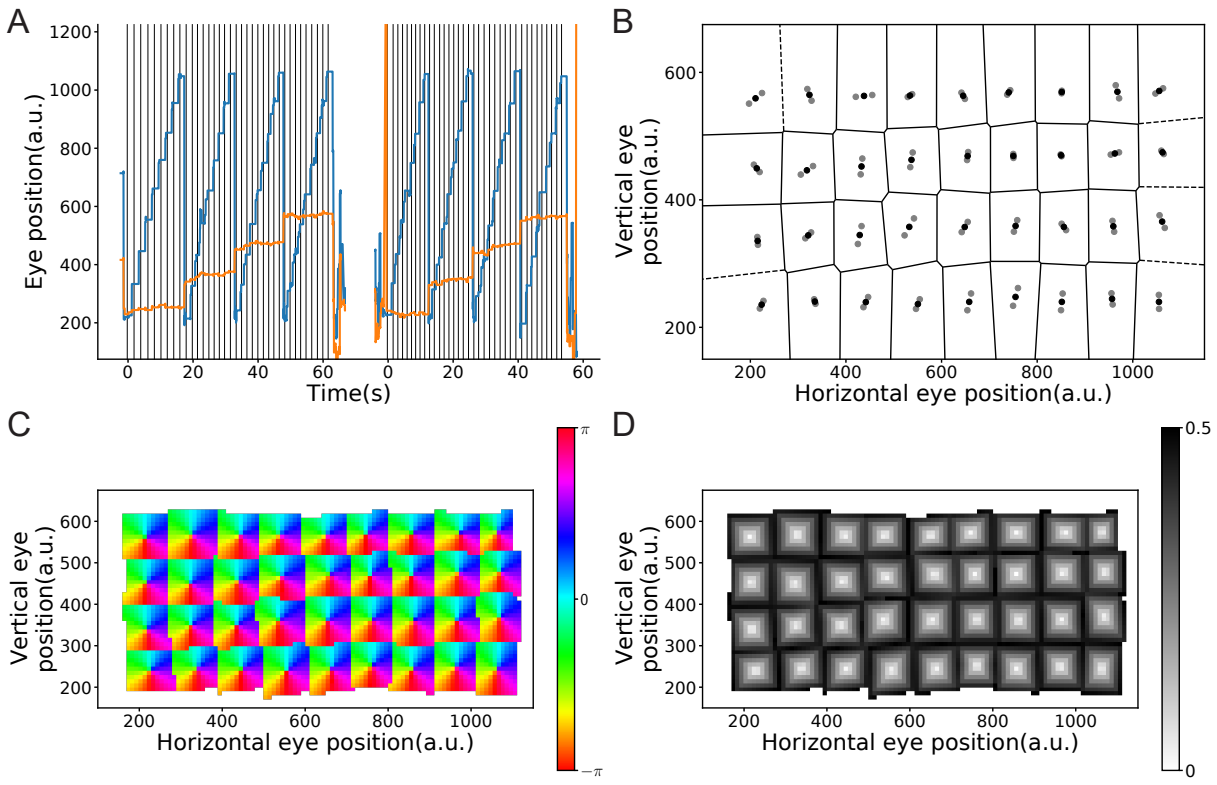

Figure 17: Eye calibration procedure. **A.** Behavioral and eye tracking data during the calibration stages before and after the experiment. The blue and orange lines indicate the horizontal and vertical eye position in eye tracker coordinates (arbitrary units), and each vertical line indicates a time at which the participant pressed the space bar. For both calibration periods, we measure time relative to the time of the first space bar press. **B.** Eye coordinates at the time of each space bar press in both calibration stages (gray dots), and a Voronoi tesselation (black solid & dashed lines) with centroids (black dots) at the average of the two calibration coordinates for each tile. **C,D.** For each point in eye coordinate space, we compute fractional board coordinates using equation 8 and display the angle formed by x and y (**C**, in hue) as well as the distance from the square center (**D**, using $d = \max(|x|, |y|)$, in grayscale). The regions in eye tracker space that lie outside the board edges is colored white. Note the correspondence between the Voronoi tesselation centroids and edges with the pinwheels and discontinuities in these maps.

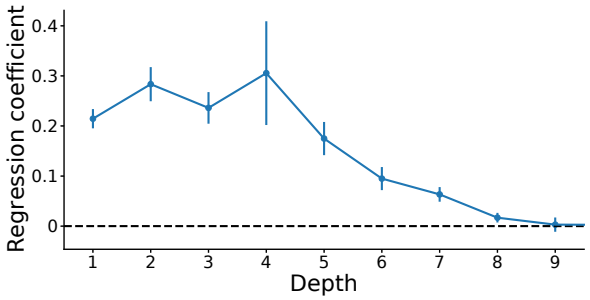

Figure 18: Coefficients in a linear regression predicting participants' attentional distribution from the distribution of squares that the model includes in its principal variation at each depth. Error bars denote mean and standard error across participants.

positions from the eye tracking data, selected using the same method as in S7, in which the **No feature drop** model assigns low probability to the participant's move. Upon inspection, we notice that these are all positions in which an immediate 4-in-a-row is possible for the participant or their opponent, and the model's preferred move is to either complete this threat, or block it. However, the participant fails to do so. On the right column, Fig S19 shows the participant's eye movements while contemplating their move in these positions. Indeed, in 4 out of 5 positions, participants spend no time whatsoever looking at the square preferred by the model, and in the remaining one (4th row), the participant looks at the relevant square but it is not their main focus of attention. Thus, this analysis suggests that positions in which the model identifies dropped features are likely instances where the participant failed to spot a particular pattern on the board.

12 Experience-dependent performance gains are mediated by metrics

To demonstrate that the derived metrics of planning depth, feature drop rate and heuristic quality are sufficient to capture experience-dependent changes in playing strength, we perform a linear regression predicting the Elo rating of participants in the learning experiment from the derived

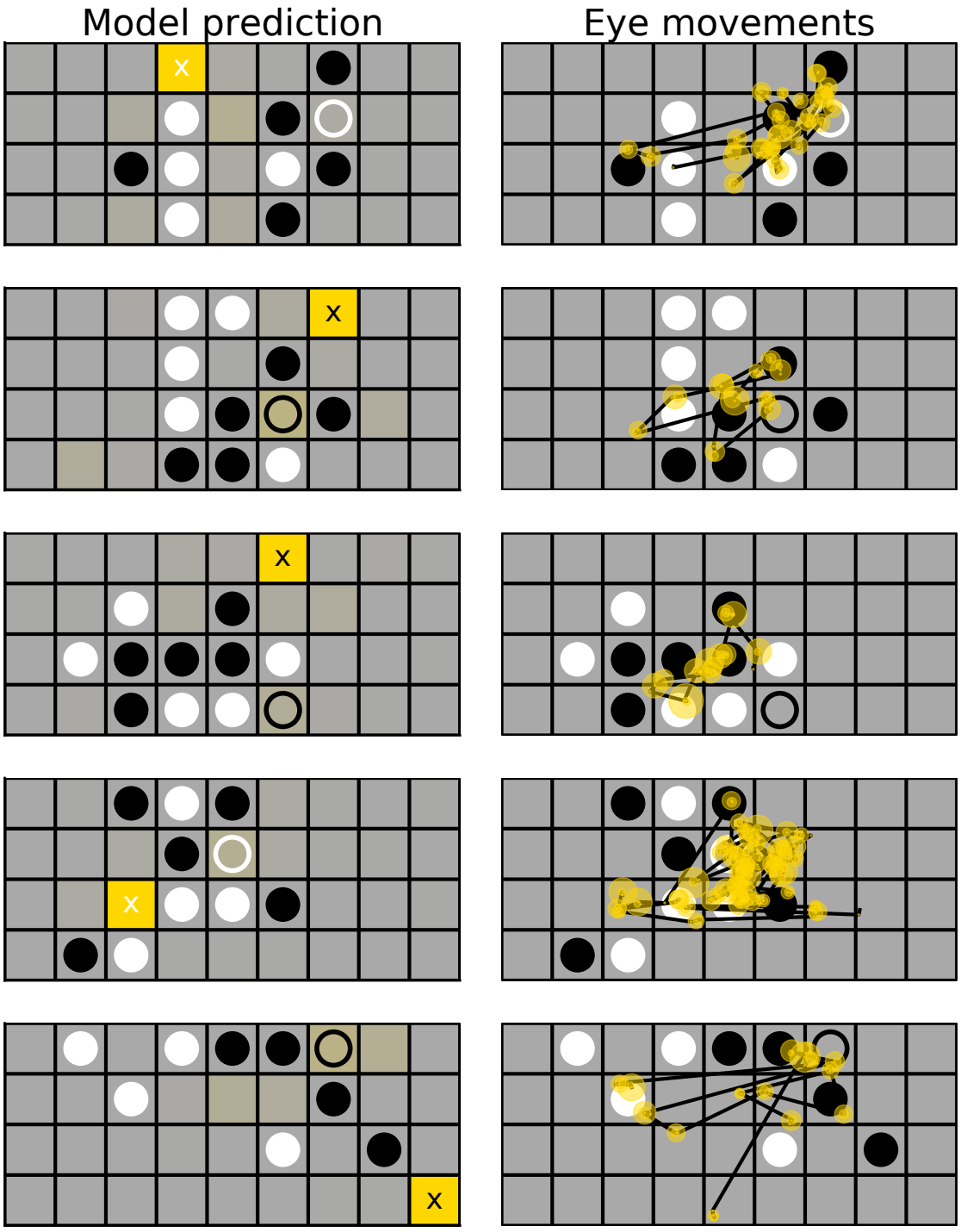

Figure 19: Same as Fig S9, but only showing the main model, and using the probability of the human move (circle) as the metric (red shading).

metrics. Specifically, for each session of the learning experiment, we perform a multiple linear regression predicting Elo rating from planning depth, feature drop rate and heuristic quality. This results in regression slopes of -818 ± 108 Elo points per feature drop rate (mean and s.e.m. across sessions), 33.9 ± 4.9 points per planning depth, and 326 ± 156 per heuristic quality. We then combine the predictions of the regression across all sessions, and show that using the derived metrics, we can explain 56.7% of the variance in participants' Elo rating (Fig S20A). Additionally, even though the regression coefficients are based on within-session data only, the observed trend of increasing playing strength across sessions is well captured by this regression (Fig S20B). Note that in this regression, we use a constant intercept for all sessions. Finally, by multiplying the regression slopes with the empirically measured changes in the derived metrics, we can break down the increase in Elo rating as 36 ± 11 points from planning, a 46 ± 12 due to featured dropping, and -6.6 ± 3.5 due to heuristic quality.

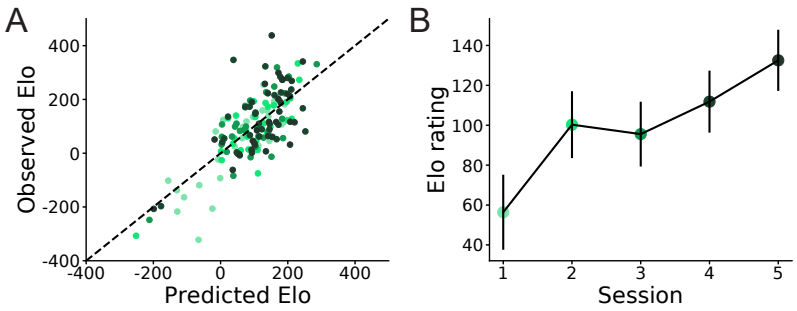

Figure 20: **A.** Predicted vs observed Elo ratings in a linear regression predicting Elo rating from derived metrics of planning depth, feature drop rate and heuristic quality. **B.** Predicted change in Elo rating across sessions using coefficients from the regression in **A.**, combined with the empirical change in the derived metrics (Fig 3).

13 Individual differences in playing strength

In Fig S18, we show the correlation between Elo ratings and the three metrics across all participants in all sessions of the learning and time pressure experiments. Playing strength correlates

with planning depth ($\rho = 0.62, p < 0.001$) and feature drop rate ($\rho = -0.73, p < 0.001$), but not with heuristic quality ($\rho = 0.11, p = 0.088$).

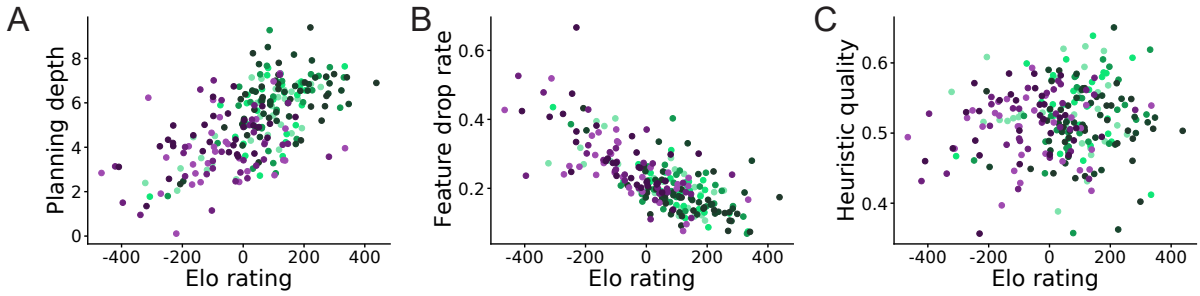

Figure 21: **A.** Planning depth vs Elo rating of all participants in the learning (green) and time pressure experiments (purple). **B.-C.** Same, for feature drop rate and heuristic quality.

14 Response times in the learning experiment

Fig S22A shows the average response time for each session in the learning experiment. Participants play slightly faster in later sessions. Therefore, our finding of increased planning in later sessions is not confounded by increases in thinking time. Instead, people plan more while using less time.

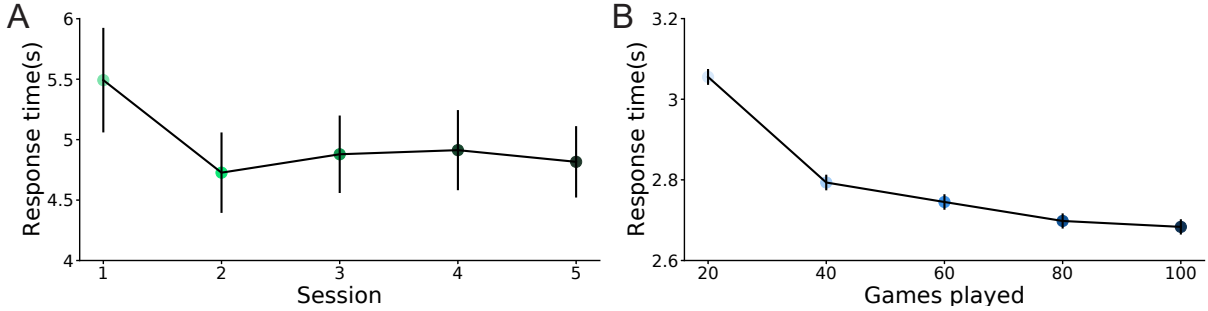

Figure 22: **A.** Response time for participants in each session of the learning experiment. Error bars denote mean and standard error across all positions encountered by all participants. **B.** Same, for users in each block of games in the mobile experiment.

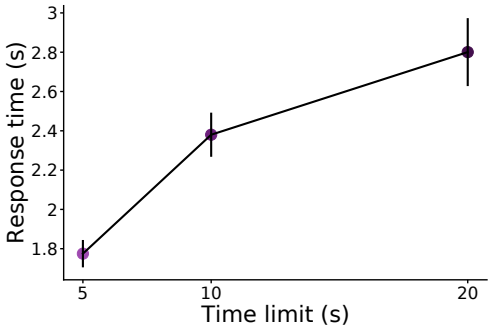

Figure 23: Response time for participants in each time limit condition in the time pressure experiment. Error bars denote mean and standard error across all positions encountered by all users.

15 Response times in the learning experiment

Fig S23 shows the average response time for each time limit condition in the time pressure experiment. The time limit manipulation is effective at increasing participants' response times, even though they use only a fraction of the available time on average.

16 Memory and reconstruction experiment

To better understand the mechanism behind participants' increase in performance and planning depth, we conducted a memory and reconstruction experiment, akin to de Groot[61], Chase & Simon[2] (See methods1.9). Table S25 shows the probability for experts and novices to correctly reconstruct a piece given the condition (real games vs randomly scrambled boards) and the identity of the piece (black, white or an empty square). A mixed effects regression shows that experts are more often correct than novices ($\beta = 0.015 \pm 0.004, p < 0.001$), and that pieces from real games are more often correctly reconstructed than random positions ($\beta = 0.058 \pm 0.005, p < 0.001$), but we find no evidence for an interaction ($\beta = 0.0002 \pm 0.0051, p = 0.96$).

			Empty	Black	White
Experts	Games	Empty	0.95	0.03	0.02
		Black	0.21	0.72	0.07
		White	0.23	0.08	0.68
	Random	Empty	0.92	0.04	0.04
		Black	0.30	0.64	0.06
		White	0.34	0.08	0.57
Novices	Games	Empty	0.95	0.03	0.02
		Black	0.19	0.75	0.06
		White	0.20	0.06	0.74
	Random	Empty	0.91	0.04	0.05
		Black	0.25	0.72	0.06
		White	0.32	0.08	0.61

Table 25: Reconstruction probabilities in the memory and reconstruction experiment. Each row indicates the probability distribution of the identity of the reconstructed piece (black, white or empty) for a given piece identity, condition and participant group.

Figure S24A shows a visual summary of this data using three error rate metrics. First, the missed piece error rate is the probability for a participant to fail to place a piece on a square during reconstruction given that a piece was present. Second, the extra piece error rate is the probability for a participant to place a piece on a square that was originally empty. Finally, and the wrong color error is the probability to place a piece of the wrong color, given that the square was originally occupied. Although experts are slightly worse than novices in the extra piece error rate ($\beta = 0.0071 \pm 0.0031, p = 0.049$), experts substantially outperform novices in the missed piece ($\beta = 0.037 \pm 0.006, p < 0.001$) and the wrong color rate ($\beta = 0.019 \pm 0.003, p < 0.001$). However, Figure S24B shows that experts also take more time to reconstruct pieces ($\beta = 2.7 \pm 0.57, p < 0.001$), meaning that the performance result could reflect a speed-accuracy trade-off as opposed to an overall improvement.

To gain more insight into the relationship between reconstruction speed, accuracy and expertise, we investigate how quickly and accurately participants reconstruct game-relevant features, in particular the connected 2-in-a-row, unconnected 2-in-a-row and 3-in-a-row feature

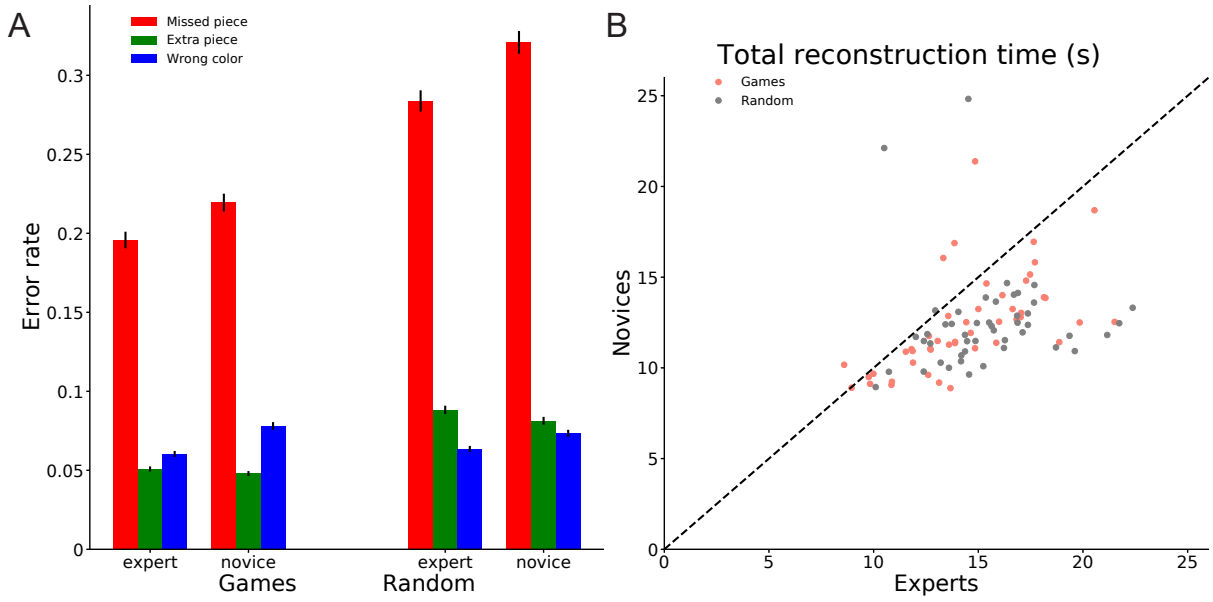

Figure 24: **A.** Error rates in the memory and reconstruction experiment. **B.** Scatterplot of total reconstruction time for experts and novices. Each point represents a board position in the memory in reconstruction experiment, the x-coordinate the average time that experts take to finish their reconstruction, and the y-coordinate the same but for novices. Positions from games are colored pink, randomly scrambled positions in gray.

(figure S25). We define the accuracy as the probability for a feature instance to be present in a given location and orientation in the reconstructed board given that it was present on the original. Since participants place piece in the reconstruction sequentially, we define the reconstruction time for a given feature instance as the time from the start of reconstruction to the first time that the feature instance is present on the reconstructed board. Note that this metric does not take into account the possibility that the participant subsequently removes those pieces, since participants rarely remove pieces ($7.3 \pm 0.5\%$ of clicks).

Figure S25 shows that experts are more accurate at reconstructing game-relevant features (3-in-a-row: $\beta = 0.062 \pm 0.020, p = 0.0046$; connected 2-in-a-row: $\beta = 0.053 \pm 0.015, p < 0.001$; unconnected 2-in-a-row: $\beta = 0.068 \pm 0.015, p < 0.001$). Additionally, when experts reconstruct these features, they at an earlier point in the trial (3-in-a-row: $\beta = 0.034 \pm 0.012, p < 0.001$; connected 2-in-a-row: $\beta = 0.028 \pm 0.008, p < 0.001$; unconnected 2-in-a-row: $\beta = 0.039 \pm 0.009, p < 0.001$), leading to an approximately equal first reconstruction time (3-in-a-row: $\beta = -0.06 \pm 0.17, p = 0.69$; connected 2-in-a-row: $\beta = 0.23 \pm 0.14, p = 0.028$; unconnected 2-in-a-row: $\beta = 0.14 \pm 0.15, p = 0.20$). In other words, experts are more accurate at reconstructing features in the same amount of time, and experts start their reconstructions with the game-relevant features. Figure S26) shows an example position in which experts reconstruct a 3-in-a-row feature more accurately than novices. Together, these results suggest that players represent boards in memory in terms of game-relevant features.

17 Robustness

To demonstrate that the main results on expertise are robust to choices in the model specification, we repeat the analysis for all alternative models. This requires extending the definition of the planning depth, feature drop rate and heuristic quality metrics to the alternative models. This is straightforward for all models except the **Orientation-dependent dropping** and

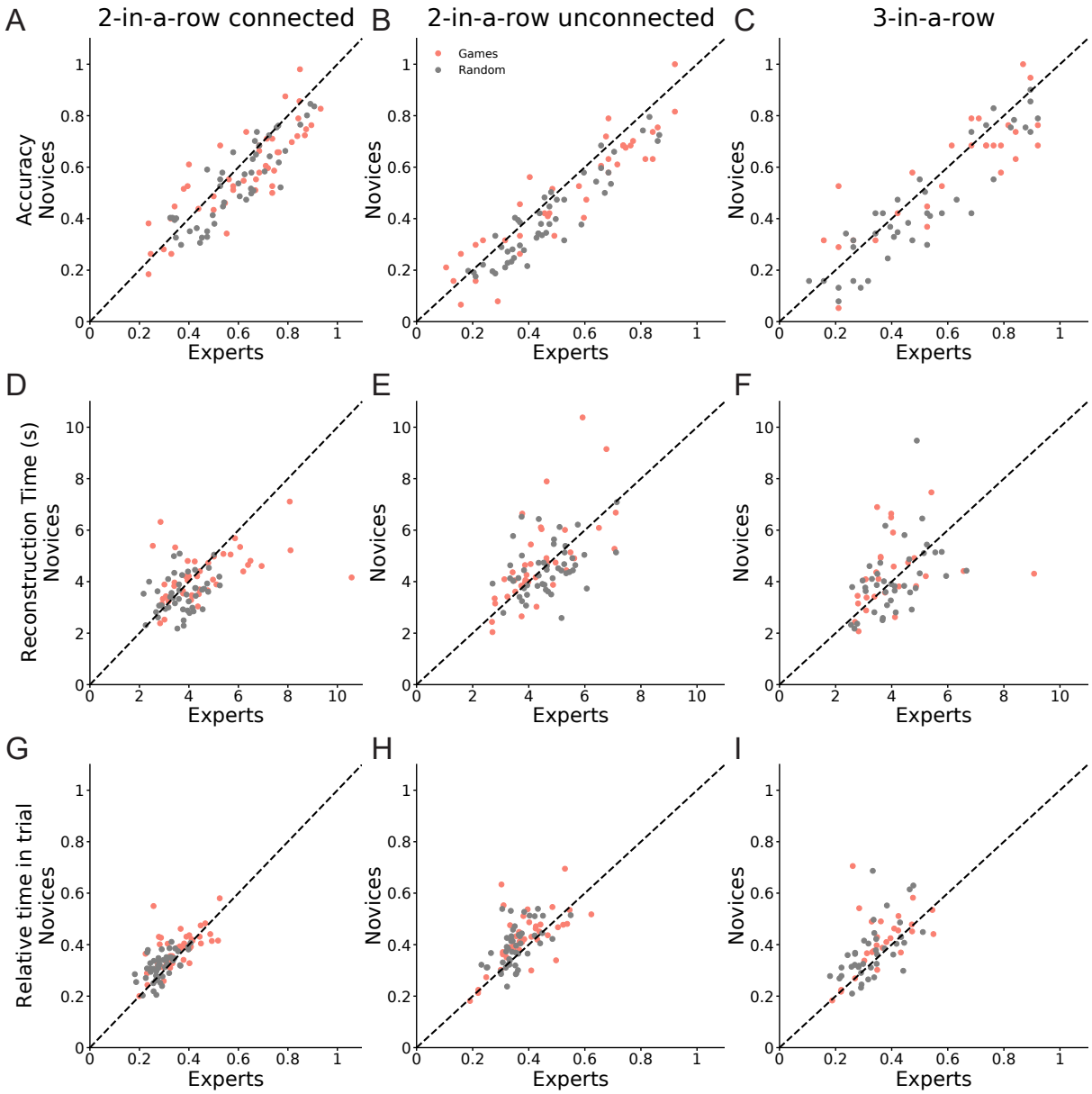

Figure 25: **A.-C.** Reconstruction accuracy in the memory experiment. Each panel shows a scatterplot for a different feature, were each point represents a board position which contains one or more features of that type, the x-coordinate the average probability for experts to correctly reconstruct those features, and the y-coordinate the same but for novices. Positions from games are colored pink, randomly scrambled positions in gray. **D.-F.** Same, for the time of first reconstruction. **G.-I.** Same, for the time of first reconstruction relative to the total trial duration.

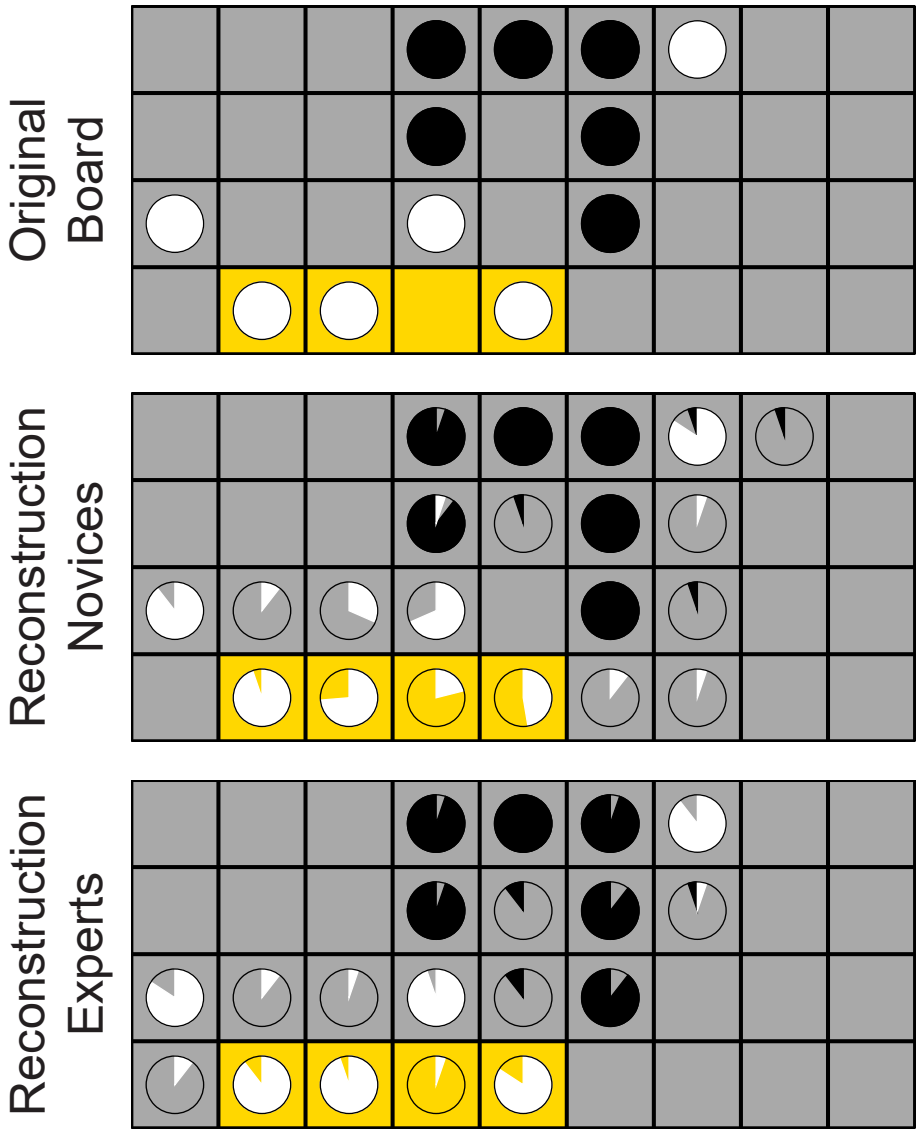

Figure 26: Example position of the memory and reconstruction experiment. The original board contains a 3-in-a-row feature on the bottom row (yellow shading). In the reconstructions, each circle indicates the distribution of pieces placed by different observers, with the angles of the gray, black and white wedges indicating the probability for that square to be empty, contain a black or contain a white piece, respectively. Novices correctly reconstruct the 3-in-a-row feature 42.1% of the time, but experts 84.2%.

Type-dependent dropping models, for which we define the feature drop rate as the drop rate of the horizontal 3-in-a-row feature. Additionally, we note that in the **Fixed depth** model, every branch of the decision tree is explored up to the same depth, hence the planning depth is not just the length of the principal variation, but also the length of every other variation.

Table S26 shows the result of a correlation between Elo rating and planning depth, feature drop rate or heuristic quality across individuals in the learning experiment (analogous to S13) for all models. Across all 22 models for which it is applicable, participants' Elo rating correlates strongly with planning depth and feature drop rate.

18 Response times in mobile data

As in the in-lab data, we verify that the increase in users' planning depth is not a result of slower play in the mobile data set. Fig S22B shows that this indeed the case, as users' response time decreases with experience.

	Loglik per move	Planning depth		Feature drop rate		Heuristic quality	
		ρ	p	ρ	p	ρ	p
Main	-1.95	0.61	$1.8 \cdot 10^{-16}$	-0.66	$3.3 \cdot 10^{-20}$	0.02	0.85
No value noise	-2.06	0.31	$1.3 \cdot 10^{-4}$	-0.38	$1.2 \cdot 10^{-6}$	0.35	$1.1 \cdot 10^{-5}$
No feature drop	-2.0	0.53	$4.7 \cdot 10^{-12}$	N/A		-0.02	0.82
No pruning	-2.0	0.71	$4.1 \cdot 10^{-24}$	-0.59	$1.7 \cdot 10^{-15}$	-0.02	0.78
No tree	-1.98	N/A		-0.59	$1.6 \cdot 10^{-15}$	-0.10	0.20
No active scaling	-1.96	0.58	$5.0 \cdot 10^{-15}$	-0.63	$7.5 \cdot 10^{-18}$	-0.05	0.56
No 3-in-a-row	-2.21	0.67	$4.3 \cdot 10^{-21}$	-0.39	$9.4 \cdot 10^{-7}$	0.19	0.02
No connected 2-in-a-row	-2.04	0.67	$8.6 \cdot 10^{-21}$	-0.51	$1.9 \cdot 10^{-11}$	-0.15	0.06
No center	-2.03	0.58	$7.9 \cdot 10^{-15}$	-0.66	$4.0 \cdot 10^{-20}$	-0.04	0.64
No 4-in-a-row	-1.98	0.65	$2.0 \cdot 10^{-19}$	-0.64	$1.2 \cdot 10^{-18}$	0.02	0.79
No unconnected 2-in-a-row	-1.97	0.59	$1.4 \cdot 10^{-15}$	-0.65	$5.0 \cdot 10^{-19}$	0.07	0.39
Optimal weights	-2.01	0.44	$1.4 \cdot 10^{-8}$	-0.61	$1.4 \cdot 10^{-16}$	N/A	
Tile dropping	-1.95	0.62	$2.2 \cdot 10^{-17}$	-0.67	$4.2 \cdot 10^{-21}$	0.02	0.85
Fixed iterations	-1.95	0.64	$2.3 \cdot 10^{-18}$	-0.31	$1.1 \cdot 10^{-4}$	0.02	0.80
Fixed depth	-1.94	0.59	$2.8 \cdot 10^{-15}$	-0.70	$1.8 \cdot 10^{-23}$	0.01	0.95
Fixed branching	-1.94	0.61	$6.7 \cdot 10^{-17}$	-0.36	$7.8 \cdot 10^{-6}$	-0.04	0.65
Orientation-dep. weights	-1.95	0.56	$7.6 \cdot 10^{-14}$	-0.64	$1.4 \cdot 10^{-18}$	-0.01	0.92
Orientation-dep. dropping	-1.95	0.61	$2.1 \cdot 10^{-16}$	-0.59	$1.6 \cdot 10^{-15}$	0.08	0.32
Triangle	-1.95	0.39	$7.5 \cdot 10^{-7}$	-0.72	$5.5 \cdot 10^{-25}$	-0.21	0.01
Type-dep. dropping	-1.95	0.24	$2.6 \cdot 10^{-3}$	-0.70	$1.7 \cdot 10^{-23}$	-0.08	0.32
Opponent scaling	-1.95	0.59	$2.3 \cdot 10^{-15}$	-0.60	$3.0 \cdot 10^{-16}$	-0.26	$1.4 \cdot 10^{-3}$

Table 26: Results of a Pearson correlation test between Elo rating and derived metrics across all participants and sessions in the learning experiment, with derived metrics computed for each model specification. The agreement in the correlation coefficients shows that our main result on the nature of expertise is robust to the choice of model specification.